القسم الأول:
برمجة القوائم في دلفي

<-----> Objects12

الملف النصي لبرنامج

(Menu2)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<-----> Object Form1: TForm1
  Left = 144
  Top = 110
  Width = 467
  Height = 341
  Caption = 'Menu Demo'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  PixelsPerInch = 96
  TextHeight = 13
  <-----> Object Panel1: TPanel
    Left = 56
    Top = 56
    Width = 345
    Height = 209
    TabOrder = 0
    <-----> Object Button1: TButton
      Left = 32
      Top = 16
      Width = 110
      Height = 25
      Caption = 'Create Main Menu'
      TabOrder = 0
      OnClick = Button1Click
    end
    <-----> Object Button2: TButton
      Left = 200
      Top = 14
      Width = 110
      Height = 25
      Caption = 'Create PopUp Menu'
      TabOrder = 1

   OnClick = Button2Click
    end
    <-----> Object Button3: TButton
      Left = 200
      Top = 52
      Width = 110
      Height = 25
      Caption = 'Add PopUp Items'
      Enabled = False
      TabOrder = 2
      OnClick = Button3Click
    end
    <-----> Object Button4: TButton
      Left = 32
      Top = 54
      Width = 110
      Height = 25
      Caption = 'Add Main Items'
      Enabled = False
      TabOrder = 3
      OnClick = Button4Click
    end
    <-----> Object Button5: TButton
      Left = 32
      Top = 92
      Width = 110
      Height = 25
      Caption = 'Add Sub-Menu'
      Enabled = False
      TabOrder = 4
      OnClick = Button5Click
    end
    <-----> Object Button6: TButton
      Left = 32
      Top = 130
      Width = 110
      Height = 25
      Caption = 'Add  Vertical Break'
      Enabled = False
      TabOrder = 5
      OnClick = Button6Click
    end
```

```
   <-----> Object Button7: TButton
     Left = 200
     Top = 90
     Width = 110
     Height = 25
     Caption = 'Show PopUp'
     Enabled = False
     TabOrder = 6
     OnClick = Button7Click
   end
   <-----> Object Button8: TButton
     Left = 32
     Top = 168
     Width = 110
     Height = 25
     Caption = 'Add Horizontal
Break'
     Enabled = False
     TabOrder = 7
     OnClick = Button8Click
   end
   <-----> Object Button9: TButton
     Left = 200
     Top = 128
     Width = 110
     Height = 25
     Caption = 'Assign OnClick
Event'
     Enabled = False
     TabOrder = 8
     OnClick = Button9Click
   end
  end
End
```

Procedures10

الملف التنفيذي لبرنامج

(Menu2)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
program PMnuDmo;

uses
  Forms,
  UMnuDmo in 'UMnuDmo.pas'
{Form1};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TForm1,
Form1);
  Application.Run;
end.

\\\\\\\\\\\\\

unit UMnuDmo;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls, ExtCtrls,Menus;

type
  TForm1 = class(TForm)
    Panel1: TPanel;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Button5: TButton;
    Button6: TButton;
    Button7: TButton;
    Button8: TButton;
    Button9: TButton;
<----->   procedure
Button1Click(Sender: TObject);
<----->   procedure
Button2Click(Sender: TObject);
<----->   procedure
Button7Click(Sender: TObject);
<----->   procedure
Button3Click(Sender: TObject);
<----->   procedure
```

```
  ShowMessage('MainMenu created but no items' + #13+
            'are added so it does not show.');
  Button1.Enabled := False;
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
  MyPopUpMenu := TPopUpMenu.Create(self);
  Button3.Enabled := true;
  ShowMessage('PopUpMenu created.');
  Button2.Enabled := False;
end;

procedure TForm1.Button7Click(Sender: TObject);
begin
  MyPopUpMenu.Popup(Form1.Left + 60, Form1.Top + 140);
  Button9.Enabled := true;
end;

procedure TForm1.Button3Click(Sender: TObject);
var
  i: Integer;
begin
  for i := 0 to 3 do begin
    MyPopUpItems[i] := TMenuItem.Create(Self);
    MyPopUpItems[i].Caption := 'New item ' + IntToStr(i);
    MyPopUpMenu.Items.Add(MyPopUpItems[i]);
  end;
  Button7.Enabled := true;
  Button3.Enabled := False;
end;
procedure TForm1.Button4Click(Sender: TObject);
MyPopupHandler(Sender: TObject);
<-----> procedure Button4Click(Sender: TObject);
<-----> procedure Button5Click(Sender: TObject);
<-----> procedure Button6Click(Sender: TObject);
<-----> procedure Button8Click(Sender: TObject);
<-----> procedure Button9Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  MyMainMenu: TMainMenu;
  MyPopUpMenu: TPopUpMenu;
  MySubItem1,MySubItem2 : TMenuItem;
  MySubItems: array[0..3] of TMenuItem;
  MyPopUpItems: array[0..3] of TMenuItem;
implementation

{$R *.DFM}

procedure TForm1.MyPopupHandler(Sender: TObject);
begin
  with Sender as TMenuItem do begin
    ShowMessage(Caption);
  end;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  MyMainMenu:= TMainMenu.Create(Self);
  Button4.Enabled := true;
```

```
TForm1.Button8Click(Sender:
TObject);
begin
  MySubItems[2].Caption :='-';
end;

procedure
TForm1.Button9Click(Sender:
TObject);
var
  i: Integer;
begin
  for i := 0 to 3 do begin
    MyPopUpItems[i].OnClick :=
MyPopupHandler;
  end;
end;

end.
```

18

الملف التنفيذي لبرنامج

(Menu)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit menudemo;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  Menus, ExtCtrls, ComCtrls;

type
  TMenuForm = class(TForm)
    DisplayPanel: TPanel;
    DemoShape: TShape;
    MainMenu: TMainMenu;
    Application1: TMenuItem;
    Exit1: TMenuItem;
    Color1: TMenuItem;
```

```
var
  MyItem: array[0..2] of
TMenuItem;
  i: Integer;
begin
  for i := 0 to 2 do begin
    MyItem[i] :=
TMenuItem.Create(Self);
    MyItem[i].Caption := 'New item '
+ IntToStr(i);

MyMainMenu.Items.Add(MyItem[i]
);
    end;
  Button4.Enabled := False;
  Button5.Enabled := true;
end;
procedure
TForm1.Button5Click(Sender:
TObject);
var
  i: Integer;
begin
  for i := 0 to 3 do begin
    MySubItems[i] :=
TMenuItem.Create(Self);
    MySubItems[i].Caption := 'New
item ' + IntToStr(i);
    MySubItems[i].OnClick :=
MyPopUpHandler;

MyMainMenu.Items[0].Add(MySub
Items[i]);
  end;
  Button6.Enabled := true;
  Button8.Enabled := true;
  Button5.Enabled := False;
end;


procedure
TForm1.Button6Click(Sender:
TObject);
begin
  MySubItems[3].Break :=
mbBarBreak;
end;

procedure
```

<-----> procedure About1Click(Sender: TObject);
<-----> procedure Random1Click(Sender: TObject);
<-----> procedure RandomizeColors1Click(Sender: TObject);
<-----> procedure InvertColors1Click(Sender: TObject);
private
{ Private declarations }
Roundable: Boolean;
function RandomColor: TColor;
<-----> procedure AlterShape(shape: TShapeType; roundable: Boolean);
public
{ Public declarations }
end;
const
SHAPE_OFFSET = 4;
var
MenuForm: TMenuForm;
implementation
uses about;
{$R *.DFM}
procedure TMenuForm.DisplayPanelResize(Sender: TObject);
begin
DemoShape.Top := SHAPE_OFFSET;
DemoShape.Left := DemoShape.Top;
DemoShape.Height := DisplayPanel.Height - 2 * SHAPE_OFFSET;
DemoShape.Width := DisplayPanel.Width - 2 * SHAPE_OFFSET;

Outline1: TMenuItem;
Randomize1: TMenuItem;
Select1: TMenuItem;
Inside1: TMenuItem;
Randomize2: TMenuItem;
Select2: TMenuItem;
Shape1: TMenuItem;
Circle1: TMenuItem;
Ellipse1: TMenuItem;
Rectangle1: TMenuItem;
Square1: TMenuItem;
Misc1: TMenuItem;
ThickOutline1: TMenuItem;
RoundedShape1: TMenuItem;
Help1: TMenuItem;
About1: TMenuItem;
SolidColorDialog: TColorDialog;
AnyColorDialog: TColorDialog;
PopupMenu: TPopupMenu;
Random1: TMenuItem;
RandomizeColors1: TMenuItem;
InvertColors1: TMenuItem;
<-----> procedure DisplayPanelResize(Sender: TObject);
<-----> procedure Exit1Click(Sender: TObject);
<-----> procedure Randomize1Click(Sender: TObject);
<-----> procedure Select1Click(Sender: TObject);
<-----> procedure Randomize2Click(Sender: TObject);
<-----> procedure Select2Click(Sender: TObject);
<-----> procedure Circle1Click(Sender: TObject);
<-----> procedure Ellipse1Click(Sender: TObject);
<-----> procedure Rectangle1Click(Sender: TObject);
<-----> procedure Square1Click(Sender: TObject);
<-----> procedure ThickOutline1Click(Sender: TObject);
<-----> procedure RoundedShape1Click(Sender: TObject);

```
begin
  DemoShape.Brush.Color :=
RandomColor;
end;

procedure
TMenuForm.Select2Click(Sender:
TObject);
begin
  AnyColorDialog.Color :=
DemoShape.Brush.Color;
  try
    if AnyColorDialog.Execute then
      DemoShape.Brush.Color :=
AnyColorDialog.Color;
  except
    ShowMessage('Color selection
dialog failed to load.');
  end;
end;

procedure ToggleCheck(Sender:
TObject);
var
  Item: TMenuItem;
begin
  Item := Sender as TMenuItem;
  Item.Checked := not Item.Checked;
end;

procedure SetCheck(Sender:
TObject);
var
  Item: TMenuItem;
begin
  Item := Sender as TMenuItem;
  Item.Checked := True;
end;

procedure
TMenuForm.AlterShape(shape:
TShapeType; roundable: Boolean);
begin
  Self.Roundable := roundable;
  DemoShape.Shape := shape;
end;

procedure
TMenuForm.Circle1Click(Sender:
end;

procedure
TMenuForm.Exit1Click(Sender:
TObject);
begin
  Close;
end;

function
TMenuForm.RandomColor;
var
  red, green, blue: Byte;
begin
  red := Random(255);
  green := Random(255);
  blue := Random(255);

  Result := red or (green shl 8) or
(blue shl 16);
end;

procedure
TMenuForm.Randomize1Click(Sen
der: TObject);
begin
  DemoShape.Pen.Color :=
RandomColor;
end;

procedure
TMenuForm.Select1Click(Sender:
TObject);
begin
  try
    SolidColorDialog.Color :=
DemoShape.Pen.Color;
    if SolidColorDialog.Execute then
      DemoShape.Pen.Color :=
SolidColorDialog.Color;
  except
    ShowMessage('Color selection
dialog failed to load');
  end;
end;

procedure
TMenuForm.Randomize2Click(Sen
der: TObject);
```

```
  ToggleCheck(Sender);
  case DemoShape.Shape of
    stRectangle:   DemoShape.Shape := stRoundRect;
    stRoundRect:   DemoShape.Shape := stRectangle;
    stSquare:     DemoShape.Shape := stRoundSquare;
    stRoundSquare: DemoShape.Shape := stSquare;
  end;
end;

procedure TMenuForm.About1Click(Sender: TObject);
var
  AboutBox: TAboutBox;
begin
  AboutBox := TAboutBox.Create(Self);
  try
    AboutBox.ShowModal;
  finally
    AboutBox.Free;
  end;
end;

procedure TMenuForm.Random1Click(Sender : TObject);
var
  newshape: TShapeType;
begin
  // Ensure we actually change the shape.
  repeat newshape := TShapeType(Random(6)) until
    newshape <> DemoShape.Shape;

  // Adjust the "Shape" and Rounded Shape" menu selections.
  case newshape of
    stEllipse: Ellipse1Click(Ellipse1);
    stCircle: Circle1Click(Circle1);
    stRectangle, stRoundRect:
      begin
        RoundedShape1.Checked := newshape = stRoundRect;
TObject);
begin
  SetCheck(Sender);
  AlterShape(stCircle, False);
end;

procedure TMenuForm.Ellipse1Click(Sender: TObject);
begin
  SetCheck(Sender);
  AlterShape(stEllipse, False);
end;

procedure TMenuForm.Rectangle1Click(Sender: TObject);
begin
  SetCheck(Sender);
  if RoundedShape1.Checked then
    AlterShape(stRoundRect, True)
  else
    AlterShape(stRectangle, True);
end;

procedure TMenuForm.Square1Click(Sender: TObject);
begin
  SetCheck(Sender);
  if RoundedShape1.Checked then
    AlterShape(stRoundSquare, True)
  else
    AlterShape(stSquare, True);
end;

procedure TMenuForm.ThickOutline1Click(Sender: TObject);
begin
  ToggleCheck(Sender);
  DemoShape.Pen.Width := 11 - DemoShape.Pen.Width;
end;

procedure TMenuForm.RoundedShape1Click(Sender: TObject);
begin
```

```
  Label2: TLabel;
  Button1: TButton;
<----->   procedure
Button1Click(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
 AboutBox: TAboutBox;

implementation

{$R *.DFM}


procedure
TAboutBox.Button1Click(Sender:
TObject);
begin
 Close;
end;

end.
\\\\\\\\\\\\\\\\\\\\
```

Object$

الملف النصي لبرنامج

(Menu)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
object MenuForm: TMenuForm
 Left = 183
 Top = 115
 Width = 696
 Height = 480
 Caption = 'MenuForm'
 Color = clBtnFace
 Font.Charset =
```

```
      Rectangle1Click(Rectangle1);
    end;
   stSquare, stRoundSquare:
    begin
      RoundedShape1.Checked :=
newshape = stRoundSquare;
      Square1Click(Square1);
    end;
   end;
end;

procedure
TMenuForm.RandomizeColors1Clic
k(Sender: TObject);
begin
 DemoShape.Brush.Color :=
RandomColor;
 DemoShape.Pen.Color :=
RandomColor;
end;

procedure
TMenuForm.InvertColors1Click(Se
nder: TObject);
var
 i : Integer;
begin
 i :=
Integer(DemoShape.Brush.Color)
xor $FFFFFF;
 DemoShape.Brush.Color :=
TColor(i);
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\
unit about;

interface

uses
 Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
 StdCtrls;

type
 TAboutBox = class(TForm)
  Label1: TLabel;
```

```
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  Menu = MainMenu
  OldCreateOrder = True
  Position = poDefaultPosOnly
  ShowHint = True
  Visible = True
  PixelsPerInch = 96
  TextHeight = 13
  object DisplayPanel: TPanel
    Left = 0
    Top = 0
    Width = 688
    Height = 426
    Align = alClient
    BevelInner = bvLowered
    BevelOuter = bvLowered
    PopupMenu = PopupMenu
    TabOrder = 0
    OnResize = DisplayPanelResize
    object DemoShape: TShape
      Left = 120
      Top = 8
      Width = 193
      Height = 217
      Shape = stCircle
    end
  end
  object MainMenu: TMainMenu
    Left = 56
    Top = 16
    object Application1: TMenuItem
      Caption = '&Application'
      Hint = 'Simple command menu
item'
      object Exit1: TMenuItem
        Caption = 'E&xit'
        Hint = 'Close the application'
        OnClick = Exit1Click
      end
    end
    object Color1: TMenuItem
      Caption = '&Color'
      Hint = 'Submenus'
      object Outline1: TMenuItem
        Caption = '&Outline'
        Hint = 'Adjust shape outline
color'
        object Randomize1:
TMenuItem
          Caption = '&Randomize'
          Hint = 'Set shape outline to a
random color'
          OnClick = Randomize1Click
        end
        object Select1: TMenuItem
          Caption = '&Select...'
          Hint = 'Select shape outline
color'
          OnClick = Select1Click
        end
      end
      object Inside1: TMenuItem
        Caption = '&Inside'
        Hint = 'Adjust shape inside
color'
        object Randomize2:
TMenuItem
          Caption = '&Randomize'
          Hint = 'Set shape inside to a
random color'
          OnClick = Randomize2Click
        end
        object Select2: TMenuItem
          Caption = '&Select...'
          Hint = 'Select a color for the
shape filling'
          OnClick = Select2Click
        end
      end
    end
    object Shape1: TMenuItem
      Caption = '&Shape'
      GroupIndex = 1
      Hint = 'Mutually-exclusive toggle
group'
      object Circle1: TMenuItem
        Caption = '&Circle'
        Checked = True
        GroupIndex = 1
        Hint = 'Make the shape a circle'
        RadioItem = True
        OnClick = Circle1Click
```

```
RoundedShape1Click
    end
  end
  object Help1: TMenuItem
    Caption = '&Help'
    GroupIndex = 1
    Hint = 'Simple command menu
item'
    object About1: TMenuItem
      Caption = '&About...'
      Hint = 'Summons an About box'
      OnClick = About1Click
    end
  end
 end
 object SolidColorDialog:
TColorDialog
  Options = [cdSolidColor]
  Left = 56
  Top = 72
 end
 object AnyColorDialog:
TColorDialog
  Options = [cdAnyColor]
  Left = 56
  Top = 208
 end
 object PopupMenu: TPopupMenu
  Left = 64
  Top = 144
  object Random1: TMenuItem
    Caption = 'Randomize &Shape'
    OnClick = Random1Click
  end
  object RandomizeColors1:
TMenuItem
    Caption = 'Randomize &Colors'
    OnClick =
RandomizeColors1Click
  end
  object InvertColors1: TMenuItem
    Caption = '&Invert Inside Color'
    OnClick = InvertColors1Click
  end
 end
End

object AboutBox: TAboutBox
```

```
    end
    object Ellipse1: TMenuItem
      Caption = '&Ellipse'
      GroupIndex = 1
      Hint = 'Make the shape an
ellipse'
      RadioItem = True
      OnClick = Ellipse1Click
    end
    object Rectangle1: TMenuItem
      Caption = '&Rectangle'
      GroupIndex = 1
      Hint = 'Make the shape a
rectangle'
      RadioItem = True
      OnClick = Rectangle1Click
    end
    object Square1: TMenuItem
      Caption = '&Square'
      GroupIndex = 1
      Hint = 'Make the shape a
square'
      RadioItem = True
      OnClick = Square1Click
    end
  end
  object Misc1: TMenuItem
    Caption = '&Misc'
    GroupIndex = 1
    Hint =
      'Simple checkable menu items
with a superfluous bar break in
betw' +
      'een'
    object ThickOutline1:
TMenuItem
      Caption = '&Thick Outline'
      Hint = 'Controls whether or not
the outline is thick'
      OnClick = ThickOutline1Click
    end
    object RoundedShape1:
TMenuItem
      Break = mbBarBreak
      Caption = '&Rounded Shape'
      Hint = 'Makes rectangle or
square have rounded corners'
      OnClick =
```

```
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

Procedures5

الملف التنفيذي لبرنامج

(TabCntrl)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
program clsexm;

uses
  Forms,
  clsexm1 in 'clsexm1.pas' {Form1};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TForm1,
Form1);
  Application.Run;
end.
\\\\\\\\\\\\\\\
unit clsexm1;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  Db, DBTables, ComCtrls, ExtCtrls,
DBCtrls, StdCtrls, Mask;

type
  TForm1 = class(TForm)
    TabControl1: TTabControl;
    Table1: TTable;
    DataSource1: TDataSource;
    Table1EmpNo: TIntegerField;
    Table1LastName: TStringField;
```

```
  Left = 189
  Top = 110
  BorderIcons = []
  BorderStyle = bsDialog
  Caption = 'About Menu'
  ClientHeight = 134
  ClientWidth = 239
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  PixelsPerInch = 96
  TextHeight = 13
  object Label1: TLabel
    Left = 60
    Top = 24
    Width = 123
    Height = 16
    Alignment = taCenter
    Caption = 'Menu Demonstration'
    Font.Charset = ANSI_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
  end
  object Label2: TLabel
    Left = 32
    Top = 56
    Width = 177
    Height = 13
    Caption = 'Copyright (c) 1999
Inprise Corporation'
  end
  object Button1: TButton
    Left = 80
    Top = 88
    Width = 75
    Height = 25
    Caption = 'OK'
    TabOrder = 0
    OnClick = Button1Click
```

```
    begin
 
TabControl1.Tabs.Add(FieldByName('LASTNAME').AsString);
      Next;
    end;
    // Sync page and table.
    First;
  end;
end;

<-----> procedure TForm1.TabControl1Change(Sender: TObject);
begin
  with TabControl1 do
  begin
    if TabIndex <> -1 then
      if not Table1.Locate('LASTNAME', Tabs[TabIndex], []) then
        ShowMessage(Tabs[TabIndex - 1] + ' not found.');
  end;
end;

<-----> procedure TForm1.Table1CalcFields(DataSet: TDataSet);
begin
  DataSet['FullName'] := DataSet['LASTNAME'] + ', ' + DataSet['FIRSTNAME'];
end;

end.

\\\\\\
```

```
    Table1FirstName: TStringField;
    Table1PhoneExt: TStringField;
    Table1HireDate: TDateTimeField;
    Table1Salary: TFloatField;
    Label1: TLabel;
    DBEdit1: TDBEdit;
    Label2: TLabel;
    DBEdit2: TDBEdit;
    Label3: TLabel;
    DBEdit3: TDBEdit;
    Label4: TLabel;
    DBEdit4: TDBEdit;
    Label5: TLabel;
    DBEdit5: TDBEdit;
    Label6: TLabel;
    DBEdit6: TDBEdit;
    DBText1: TDBText;
    Table1FullName: TStringField;
    <-----> procedure FormCreate(Sender: TObject);
    <-----> procedure TabControl1Change(Sender: TObject);
    <-----> procedure Table1CalcFields(DataSet: TDataSet);
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

<-----> procedure TForm1.FormCreate(Sender: TObject);
begin
  TabControl1.Align := alClient;
  TabControl1.TabHeight := 30;
  with Table1 do
  begin
    Close;
    IndexName := 'ByName';
    Open;
    while not EOF do
```

الملف النصي لبرنامج

(TabCntrl)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
object Form1: TForm1
  Left = 292
  Top = 153
  Width = 366
  Height = 354
  Caption = 'Form1'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  object TabControl1: TTabControl
    Left = 0
    Top = 0
    Width = 358
    Height = 320
    Align = alClient
    TabOrder = 0
    OnChange = TabControl1Change
    object Label1: TLabel
      Left = 16
      Top = 72
      Width = 35
      Height = 13
      Caption = 'EmpNo'
      FocusControl = DBEdit1
    end
    object Label2: TLabel
      Left = 16
      Top = 112
      Width = 48
      Height = 13
      Caption = 'LastName'
      FocusControl = DBEdit2
    end
    object Label3: TLabel
      Left = 16
      Top = 152
      Width = 47
      Height = 13
      Caption = 'FirstName'
      FocusControl = DBEdit3
    end
    object Label4: TLabel
      Left = 16
      Top = 192
      Width = 46
      Height = 13
      Caption = 'PhoneExt'
      FocusControl = DBEdit4
    end
    object Label5: TLabel
      Left = 16
      Top = 232
      Width = 42
      Height = 13
      Caption = 'HireDate'
      FocusControl = DBEdit5
    end
    object Label6: TLabel
      Left = 16
      Top = 272
      Width = 29
      Height = 13
      Caption = 'Salary'
      FocusControl = DBEdit6
    end
    object DBText1: TDBText
      Left = 56
      Top = 48
      Width = 6
      Height = 20
      AutoSize = True
      DataField = 'FullName'
      DataSource = DataSource1
      Font.Charset = ANSI_CHARSET
      Font.Color = clBlack
      Font.Height = -16
```

```
  end
  object DBEdit6: TDBEdit
    Left = 16
    Top = 288
    Width = 64
    Height = 21
    DataField = 'Salary'
    DataSource = DataSource1
    TabOrder = 5
  end
 end
 object Table1: TTable
  Active = True
  OnCalcFields = Table1CalcFields
  DatabaseName = 'DBDEMOS'
  TableName = 'EMPLOYEE.DB'
  Left = 16
  Top = 16
  object Table1EmpNo:
TIntegerField
    FieldName = 'EmpNo'
    DisplayFormat = 'Emp''#'' 0000'
    MaxValue = 9999
    MinValue = 1
  end
  object Table1LastName:
TStringField
    FieldName = 'LastName'
  end
  object Table1FirstName:
TStringField
    FieldName = 'FirstName'
    Size = 15
  end
  object Table1PhoneExt:
TStringField
    FieldName = 'PhoneExt'
    Size = 4
  end
  object Table1HireDate:
TDateTimeField
    FieldName = 'HireDate'
  end
  object Table1Salary: TFloatField
    FieldName = 'Salary'
  end
  object Table1FullName:
TStringField
```

```
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentFont = False
  end
  object DBEdit1: TDBEdit
    Left = 16
    Top = 88
    Width = 64
    Height = 21
    DataField = 'EmpNo'
    DataSource = DataSource1
    TabOrder = 0
  end
  object DBEdit2: TDBEdit
    Left = 16
    Top = 128
    Width = 124
    Height = 21
    DataField = 'LastName'
    DataSource = DataSource1
    TabOrder = 1
  end
  object DBEdit3: TDBEdit
    Left = 16
    Top = 168
    Width = 94
    Height = 21
    DataField = 'FirstName'
    DataSource = DataSource1
    TabOrder = 2
  end
  object DBEdit4: TDBEdit
    Left = 16
    Top = 208
    Width = 28
    Height = 21
    DataField = 'PhoneExt'
    DataSource = DataSource1
    TabOrder = 3
  end
  object DBEdit5: TDBEdit
    Left = 16
    Top = 248
    Width = 64
    Height = 21
    DataField = 'HireDate'
    DataSource = DataSource1
    TabOrder = 4
```

{$R *.res}
begin
Application.Initialize;
Application.CreateForm(TTextBrowserForm, TextBrowserForm);
Application.Run;
end.
\\
unit main;
interface
uses
SysUtils, Types, Classes, QGraphics, QControls, QForms, QDialogs,
QComCtrls, QStdCtrls, QExtCtrls, QButtons;
type
TTextBrowserForm = class(TForm)
HomeBtn: TButton;
BackBtn: TButton;
ForwardBtn: TButton;
TextBrowser: TTextBrowser;
GroupBox1: TGroupBox;
RadioButton1: TRadioButton;
RadioButton2: TRadioButton;
Label1: TLabel;
LinkSourceEdit: TEdit;
ReloadBtn: TButton;
GroupBox2: TGroupBox;
RadioButton3: TRadioButton;
RadioButton4: TRadioButton;
RadioGroup1: TRadioGroup;
UnderlineCheckbox: TCheckBox;
StatusBar: TStatusBar;
SpeedButton1: TSpeedButton;
<------> procedure HomeBtnClick(Sender: T<------->Object);
<------> procedure TextBrowserTextChanged(Sender:

FieldKind = fkCalculated
FieldName = 'FullName'
Size = 30
Calculated = True
end
end
object DataSource1: TDataSource
DataSet = Table1
Left = 56
Top = 24
end
End

القسم الثاني : برامج النصوص في دلفي

Procedures 32
الملف التنفيذي لبرنامج
(TextBrowser)
إعداد **علاء الدين اللباد**
هاتف ٠٩٤٤٥٧٥٣٧١
الملف التنفيذي للبرنامج
TextBrowser;
ويوجد فيه
Procedures 32
إعداد:
مركز علاء الدين للمعلوماتية
علاء الدين محمد اللباد
٠٩٤٤٥٧٥٣٧١
program TextBrowser;
uses
QForms,
main in 'main.pas' {TextBrowserForm};

{ Public declarations }
end;
var
TextBrowserForm: TTextBrowserForm;
implementation
{$R *.xfm}
const
DefaultDocName = 'index.html';
FormCaption = 'TTextBrowser Demo Application - ';
<------> procedure TTextBrowserForm.FormCreate(Sender: T<------->Object);
begin
PathList := TStringList.Create;
LinkSourceEdit.Text := ExtractFilePath(Application.ExeName);
TextBrowser.FileName := LinkSourceEdit.Text + DefaultDocName;
UnderlineCheckbox.Checked := not TextBrowser.UnderlineLink;
end;
<------> procedure TTextBrowserForm.HomeBtnClick(Sender: T<------->Object);
begin
TextBrowser.Home;
end;
<------> procedure TTextBrowserForm.TextBrowserTextChanged(Sender: T<------->Object);
begin
BackBtn.Enabled := TextBrowser.CanGoBackward;
ForwardBtn.Enabled := TextBrowser.CanGoForward;
Caption := FormCaption + TextBrowser.DocumentTitle;

T<------->Object);
<------> procedure BackBtnClick(Sender: T<------->Object);
<------> procedure ForwardBtnClick(Sender: T<------->Object);
<------> procedure RadioButton1Click(Sender: T<------->Object);
<------> procedure RadioButton2Click(Sender: T<------->Object);
<------> procedure LinkSourceEditChange(Sender: T<------->Object);
<------> procedure FormCreate(Sender: T<------->Object);
<------> procedure FormDestroy(Sender: T<------->Object);
<------> procedure ReloadBtnClick(Sender: T<------->Object);
<------> procedure RadioButton3Click(Sender: T<------->Object);
<------> procedure RadioButton4Click(Sender: T<------->Object);
<------> procedure RadioGroup1Click(Sender: T<------->Object);
<------> procedure UnderlineCheckboxClick(Sender: T<------->Object);
<------> procedure TextBrowserHighlightText(Sender: T<------->Object;
const HighlightedText: WideString);
<------> procedure SpeedButton1Click(Sender: T<------->Object);
private
{ Private declarations }
PathList: TStrings;
public

```
TTextBrowserForm.ReloadBtnClick
(Sender: T<------->Object);
begin

TextBrowser.LoadFromFile(TextBr
owser.FileName);
end;

<------> procedure
TTextBrowserForm.RadioButton3C
lick(Sender: T<------->Object);
begin
  TextBrowser.TextColor := clBlack;
end;

<------> procedure
TTextBrowserForm.RadioButton4C
lick(Sender: T<------->Object);
begin
  TextBrowser.TextColor := clGreen;
end;

<------> procedure
TTextBrowserForm.RadioGroup1Cl
ick(Sender: T<------->Object);
begin
  case RadioGroup1.ItemIndex of
    0: TextBrowser.TextFormat :=
tfText;
    1: TextBrowser.TextFormat :=
tfPlainText;
  end;
end;

<------> procedure
TTextBrowserForm.UnderlineChec
kboxClick(Sender: T<-------
>Object);
begin
  TextBrowser.UnderlineLink := not
TextBrowser.UnderlineLink;
end;

<------> procedure
TTextBrowserForm.TextBrowserHi
ghlightText(Sender: T<-------
>Object;
  const HighlightedText:
WideString);
```

```
end;

<------> procedure
TTextBrowserForm.BackBtnClick(S
ender: T<------->Object);
begin
  TextBrowser.Backward;
end;

<------> procedure
TTextBrowserForm.ForwardBtnCli
ck(Sender: T<------->Object);
begin
  TextBrowser.Forward;
end;

<------> procedure
TTextBrowserForm.RadioButton1C
lick(Sender: T<------->Object);
begin
  TextBrowser.LinkColor := clBlue;
end;

<------> procedure
TTextBrowserForm.RadioButton2C
lick(Sender: T<------->Object);
begin
  TextBrowser.LinkColor := clRed;
end;

<------> procedure
TTextBrowserForm.LinkSourceEdit
Change(Sender: T<------->Object);
begin

PathList.Add(LinkSourceEdit.Text);
  TextBrowser.Factory.FilePath :=
PathList;
end;


<------> procedure
TTextBrowserForm.FormDestroy(S
ender: T<------->Object);
begin
  PathList.Free;
end;

<------> procedure
```

```
  Left = 600
  Top = 16
  Width = 23
  Height = 22
  Anchors = [akTop, akRight]
  Glyph.Data = {
    }
  NumGlyphs = 2
  OnClick = SpeedButton1Click
 end
 <-------> object HomeBtn: TButton
  Left = 8
  Top = 64
  Width = 75
  Height = 25
  Caption = 'Home'
  TabOrder = 5
  OnClick = HomeBtnClick
 end
 <-------> object BackBtn: TButton
  Left = 88
  Top = 64
  Width = 75
  Height = 25
  Caption = 'Back'
  TabOrder = 0
  OnClick = BackBtnClick
 end
 <-------> object ForwardBtn:
TButton
  Left = 168
  Top = 64
  Width = 75
  Height = 25
  Caption = 'Forward'
  TabOrder = 1
  OnClick = ForwardBtnClick
 end
 <-------> object TextBrowser:
TTextBrowser
  Left = 8
  Top = 103
  Width = 624
  Height = 334
  Anchors = [akLeft, akTop,
akRight, akBottom]
  TabOrder = 2
  TextFormat = tfText
```

```
begin
  StatusBar.SimpleText := 'Link To:
' + HighlightedText;
end;

<------> procedure
TTextBrowserForm.SpeedButton1C
lick(Sender: T<------->Object);
var
  NewDir: WideString;
begin
  if SelectDirectory('Select a
Directory', '', NewDir) then
    LinkSourceEdit.Text := NewDir;
end;

end.
```

Object 18
الملف النصي للبرنامج
TextBrowse
وفيه ١٨ اجراءا
إعداد
علاء الدين اللباد
094575371

```
<-------> object TextBrowserForm:
TTextBrowserForm
  Left = 183
  Top = 131
  Width = 640
  Height = 469
  VertScrollBar.Range = 119
  HorzScrollBar.Range = 633
  ActiveControl = BackBtn
  Caption = 'TTextBrowser
Component Demo Application'
  Color = clBackground
  OnCreate = FormCreate
  OnDestroy = FormDestroy
  PixelsPerInch = 75
  TextHeight = 13
  TextWidth = 6
  <-------> object SpeedButton1:
TSpeedButton
```

```
 end
 <-------> object Label1: TLabel
   Left = 280
   Top = 1
   Width = 82
   Height = 13
   Caption = 'Link Source Path'
 end
 <-------> object LinkSourceEdit: TEdit
   Left = 280
   Top = 16
   Width = 313
   Height = 21
   Anchors = [akLeft, akTop, akRight]
   TabOrder = 6
   OnChange = LinkSourceEditChange
 end
 <-------> object ReloadBtn: TButton
   Left = 555
   Top = 64
   Width = 78
   Height = 25
   Caption = 'Reload'
   TabOrder = 7
   OnClick = ReloadBtnClick
 end
 <-------> object GroupBox2: TGroupBox
   Left = 136
   Top = 8
   Width = 129
   Height = 41
   Caption = 'Text Color'
   TabOrder = 8
   <-------> object RadioButton3: TRadioButton
     Left = 8
     Top = 16
     Width = 57
     Height = 17
     Caption = 'Black'
     Checked = True
     Font.Color = clBlack
     Font.Height = 13
```

```
   UnderlineLink = False
   OnHighlightText = TextBrowserHighlightText
   OnTextChanged = TextBrowserTextChanged
 end
 <-------> object GroupBox1: TGroupBox
   Left = 8
   Top = 8
   Width = 121
   Height = 41
   Caption = 'Link Color'
   TabOrder = 3
   <-------> object RadioButton1: TRadioButton
     Left = 8
     Top = 16
     Width = 57
     Height = 17
     Caption = 'Blue'
     Checked = True
     Font.Color = clBlue
     Font.Height = 13
     Font.Name = 'Helvetica'
     Font.Pitch = fpVariable
     Font.Style = [fsBold]
     ParentFont = False
     TabOrder = 0
     OnClick = RadioButton1Click
   end
   <-------> object RadioButton2: TRadioButton
     Left = 64
     Top = 17
     Width = 49
     Height = 17
     Caption = 'Red'
     Font.Color = clRed
     Font.Height = 13
     Font.Name = 'Helvetica'
     Font.Pitch = fpVariable
     Font.Style = [fsBold]
     ParentFont = False
     TabOrder = 1
     TabStop = False
     OnClick = RadioButton2Click
   end
```

OnClick = UnderlineCheckboxClick
end
<-------> object StatusBar: TStatusBar
Left = 0
Top = 450
Width = 640
Height = 19
Panels = <>
SimplePanel = True
SizeGrip = False
end
End

Font.Name = 'Helvetica'
Font.Pitch = fpVariable
Font.Style = [fsBold]
ParentFont = False
TabOrder = 0
OnClick = RadioButton3Click
end
<-------> object RadioButton4: TRadioButton
Left = 65
Top = 16
Width = 58
Height = 17
Caption = 'Green'
Font.Color = clGreen
Font.Height = 13
Font.Name = 'Helvetica'
Font.Pitch = fpVariable
Font.Style = [fsBold]
ParentFont = False
TabOrder = 1
TabStop = False
OnClick = RadioButton4Click
end
end
<-------> object RadioGroup1: TRadioGroup
Left = 264
Top = 48
Width = 161
Height = 52
Items.Strings = (
'Formatted'
'Plain')
Caption = 'Text Format'
ItemIndex = 0
TabOrder = 9
OnClick = RadioGroup1Click
Orientation = orHorizontal
end
<-------> object UnderlineCheckbox: TCheckBox
Left = 432
Top = 60
Width = 120
Height = 35
Caption = 'Underline Links'
TabOrder = 10

Procedures22
الملفالتنفيذي لبرنامج
(BasicEd)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
program BasicEd;

uses
  QForms,
  MainFrm in 'MainFrm.pas' {MainForm},
  AboutFrm in 'AboutFrm.pas' {AboutForm},
  EditFrm in 'EditFrm.pas' {EditForm};

begin
  Application.CreateForm(TMainForm, MainForm);
  Application.Run;
end.


\\\\\\

unit AboutFrm;

interface

uses
  SysUtils, Types, Classes, QGraphics, QControls, QForms, QDialogs,
  QStdCtrls;

type
  TAboutForm = class(TForm)
    Button1: TButton;
    Label1: TLabel;
    <------->procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

<------->procedure ShowAboutBox;

implementation

{$R *.xfm}

<------->procedure ShowAboutBox;
begin
  with TAboutForm.Create(Application) do
  begin
    ShowModal;
    Free;
  end;
end;

<------->procedure TAboutForm.Button1Click(Sender: TObject);
begin
  Close;
end;

end.

\\\\\\


unit EditFrm;

interface

uses
  SysUtils, Types, Classes, QGraphics, QControls, QForms, QDialogs,
  QStdCtrls;

type
  TEditForm = class(TForm)
    memText: TMemo;
```

```
   ToolButton5: TToolButton;
   ToolButton6: TToolButton;
   mnuFile: TMenuItem;
   mnuNew: TMenuItem;
   mnuOpen: TMenuItem;
   mnuSave: TMenuItem;
   mnuEdit: TMenuItem;
   mnuCut: TMenuItem;
   mnuCopy: TMenuItem;
   mnuPaste: TMenuItem;
   mnuExit: TMenuItem;
   mnuHelp: TMenuItem;
   mnuAbout: TMenuItem;
   OpenDialog: TOpenDialog;
   SaveDialog: TSaveDialog;
   ActionList: TActionList;
   actnCut: TEditCut;
   actnCopy: TEditCopy;
   actnPaste: TEditPaste;
   actnSelectAll: TEditSelectAll;
   actnDelete: TEditDelete;
   actnWinClose: TWindowClose;
   actnWinCascade:
 TWindowCascade;
   actnWinTile: TWindowTile;
   actnWinMinimizeAll:
 TWindowMinimizeAll;
   actnNew: TAction;
   actnSave: TAction;
   actnOpen: TAction;
   actnExit: TAction;
   mnuWindow: TMenuItem;
   mnuTile: TMenuItem;
   N1: TMenuItem;
   mnuSelectAll: TMenuItem;
   mnuMinimize: TMenuItem;
   mnuCascade: TMenuItem;
   mnuClose: TMenuItem;
   N2: TMenuItem;
   N3: TMenuItem;
   ToolButton7: TToolButton;
   ImageList: TImageList;
   <------->procedure
 mnuAboutClick(Sender: TObject);
   <------->procedure
 actnNewExecute(Sender: TObject);
   <------->procedure
 actnSaveExecute(Sender: TObject);
   <------->procedure
 FormClose(Sender: TObject; var
 Action: TCloseAction);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.xfm}

<------->procedure
TEditForm.FormClose(Sender:
TObject; var Action: TCloseAction);
begin
  Action := caFree;
end;

end.


\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\


unit MainFrm;

interface

uses
  SysUtils, Types, Classes,
QGraphics, QControls, QForms,
QDialogs,
  QExtCtrls, QComCtrls, QTypes,
QMenus, QStdCtrls, QImgList,
QActnList,
  QStdActns;

type
  TMainForm = class(TForm)
    MainMenu: TMainMenu;
    StatusBar1: TStatusBar;
    ToolBar1: TToolBar;
    ToolButton1: TToolButton;
    ToolButton2: TToolButton;
    ToolButton3: TToolButton;
    ToolButton4: TToolButton;
```

```
begin
  ShowAboutBox;
end;

<------->procedure
TMainForm.actnNewExecute(Sende
r: TObject);
begin
  CreateMDIChild(sUntitled +
IntToStr(MDIChildCount + 1));
end;

<------->procedure
TMainForm.actnSaveExecute(Sende
r: TObject);
begin
  if not Assigned(ActiveMDIChild)
then
    Exit;
  SaveDialog.FileName :=
ActiveMDIChild.Caption;
  if SaveDialog.Execute then
  begin
    if ActiveMDIChild is TEditForm
then

TEditForm(ActiveMDIChild).memT
ext.Lines.SaveToFile(SaveDialog.Fil
eName);
    ActiveMDIChild.Caption :=
SaveDialog.FileName;
  end;
end;

<------->procedure
TMainForm.actnOpenExecute(Send
er: TObject);
begin
  if OpenDialog.Execute then
  begin
    if ActiveMDIChild is TEditForm
then

TEditForm(ActiveMDIChild).memT
ext.Lines.LoadFromFile(OpenDialog
.FileName);
    ActiveMDIChild.Caption :=
OpenDialog.FileName;
  end;
```

```
    <------->procedure
actnOpenExecute(Sender: TObject);
    <------->procedure
actnExitExecute(Sender: TObject);
    <------->procedure
actnWinCloseExecute(Sender:
TObject);
    <------->procedure
FormCreate(Sender: TObject);
  private
    <------->procedure
CreateMDIChild(const Name:
string);
    { Private declarations }
  public
    { Public declarations }
  end;

var
  MainForm: TMainForm;

implementation

{$R *.xfm}

uses
  AboutFrm, EditFrm;

resourcestring
  sUntitled = 'Untitled';

<------->procedure
TMainForm.CreateMDIChild(const
Name: string);
var
  Child: TEditForm;
begin
  { create a new MDI child window }
  Child :=
TEditForm.Create(Application);
  Child.Caption := Name;
  if FileExists(Name) then
Child.MemText.Lines.LoadFromFil
e(Name);
end;

<------->procedure
TMainForm.mnuAboutClick(Sender
: TObject);
```

```
  Position = poDefault
  Scaled = False
  OnCreate = FormCreate
  PixelsPerInch = 96
  <------->Object StatusBar1: TStatusBar
    Left = 0
    Top = 339
    Width = 480
    Height = 19
    Font.CharSet = fcsLatin1
    Font.Color = clBlack
    Font.Height = 13
    Font.Name = 'Helvetica'
    Font.Pitch = fpVariable
    Font.Style = []
    Font.Weight = 40
    Panels = <>
    ParentColor = True
    ParentFont = False
  end
  <------->Object ToolBar1: TToolBar
    Left = 0
    Top = 0
    Width = 480
    Height = 29
    ButtonHeight = 23
    Flat = True
    Images = ImageList
    TabOrder = 1
    <------->Object ToolButton1: TToolButton
      HelpType = htContext
      Left = 1
      Top = 4
      Height = 23
      Action = actnNew
      Caption = '&New'
      ParentShowHint = False
      ShowHint = True
    end
    <------->Object ToolButton2: TToolButton
      HelpType = htContext
      Left = 47
      Top = 4
      Height = 23
```

```
end;

<------->procedure TMainForm.actnExitExecute(Sender: TObject);
begin
  Close;
end;

<------->procedure TMainForm.actnWinCloseExecute(Sender: TObject);
begin
  ActiveMDIChild.Close;
end;

<------->procedure TMainForm.FormCreate(Sender: TObject);
begin
  actnNewExecute(nil);
end;

end.

\\\\\\
```

<------->ObjectS54

الملف النصي لبرنامج

(BasicEd)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<------->Object MainForm: TMainForm
  Left = 183
  Top = 108
  Width = 480
  Height = 381
  VertScrollBar.Range = 48
  Caption = 'Basic Edit'
  Color = clBackground
  FormStyle = fsMDIForm
  Menu = MainMenu
```

```
    end
    <------->Object ToolButton7:
TToolButton
      HelpType = htContext
      Left = 24
      Top = 4
      Height = 23
      Action = actnOpen
      Caption = '&Open'
    end
  end
  <------->Object MainMenu:
TMainMenu
    Images = ImageList
    Left = 24
    Top = 40
    <------->Object mnuFile:
TMenuItem
      Caption = 'File'
      HelpType = htContext
      <------->Object mnuNew:
TMenuItem
        Action = actnNew
      end
      <------->Object mnuOpen:
TMenuItem
        Action = actnOpen
      end
      <------->Object mnuSave:
TMenuItem
        Action = actnSave
      end
      <------->Object N3: TMenuItem
        Caption = '-'
        HelpType = htContext
      end
      <------->Object mnuExit:
TMenuItem
        Action = actnExit
      end
    end
    <------->Object mnuEdit:
TMenuItem
      Caption = 'Edit'
      HelpType = htContext
      <------->Object mnuCut:
TMenuItem
        Action = actnCut
```

```
      Action = actnSave
      Caption = 'Save As...'
      ParentShowHint = False
      ShowHint = True
    end
    <------->Object ToolButton3:
TToolButton
      Left = 70
      Top = 4
      Width = 8
      Height = 23
      Style = tbsSeparator
      Caption = 'ToolButton3'
      ImageIndex = 2
      Wrap = True
    end
    <------->Object ToolButton4:
TToolButton
      HelpType = htContext
      Left = 78
      Top = 4
      Height = 23
      Action = actnCut
      Caption = 'Cu&t'
      ParentShowHint = False
      ShowHint = True
    end
    <------->Object ToolButton5:
TToolButton
      HelpType = htContext
      Left = 101
      Top = 4
      Height = 23
      Action = actnCopy
      Caption = '&Copy'
      ParentShowHint = False
      ShowHint = True
    end
    <------->Object ToolButton6:
TToolButton
      HelpType = htContext
      Left = 124
      Top = 4
      Height = 23
      Action = actnPaste
      Caption = '&Paste'
      ParentShowHint = False
      ShowHint = True
```

```
TMenuItem
        Caption = 'About...'
        OnClick = mnuAboutClick
        HelpType = htContext
      end
    end
  end
  <------->Object OpenDialog: TOpenDialog
    Filter = '*.txt|*.*'#10
    FilterIndex = 0
    Height = 0
    Title = 'Open a text file'
    Width = 0
    Left = 168
    Top = 40
  end
  <------->Object SaveDialog: TSaveDialog
    DefaultExt = 'txt'
    FilterIndex = 0
    Height = 0
    Title = 'Save As'
    Width = 0
    Left = 96
    Top = 40
  end
  <------->Object ActionList: TActionList
    Images = ImageList
    Left = 240
    Top = 40
    <------->Object actnNew: TAction
      Category = 'File'
      Caption = '&New'
      Hint = 'Create a new text file'
      ImageIndex = 0
      OnExecute = actnNewExecute
    end
    <------->Object actnCut: TEditCut
      Category = 'Edit'
      Caption = 'Cu&t'
      Hint = 'Cut'
      ImageIndex = 3
      ShortCut = 16472
    end
    <------->Object actnCopy: TEditCopy
      end
      <------->Object mnuCopy: TMenuItem
        Action = actnCopy
      end
      <------->Object mnuPaste: TMenuItem
        Action = actnPaste
      end
      <------->Object N1: TMenuItem
        Caption = '-'
        HelpType = htContext
      end
      <------->Object mnuSelectAll: TMenuItem
        Action = actnSelectAll
      end
    end
    <------->Object mnuWindow: TMenuItem
      Caption = 'Window'
      HelpType = htContext
      <------->Object mnuTile: TMenuItem
        Action = actnWinTile
      end
      <------->Object mnuMinimize: TMenuItem
        Action = actnWinMinimizeAll
      end
      <------->Object mnuCascade: TMenuItem
        Action = actnWinCascade
      end
      <------->Object N2: TMenuItem
        Caption = '-'
        HelpType = htContext
      end
      <------->Object mnuClose: TMenuItem
        Action = actnWinClose
      end
    end
    <------->Object mnuHelp: TMenuItem
      Caption = 'Help'
      HelpType = htContext
      <------->Object mnuAbout:
```

```
    Category = 'Window'
    Caption = '&Tile Windows'
  end
  <------->Object actnWinMinimizeAll: TWindowMinimizeAll
    Category = 'Window'
    Caption = '&Minimize All'
  end
  <------->Object actnSave: TAction
    Category = 'File'
    Caption = 'Save As...'
    Hint = 'Save the current file as...'
    ImageIndex = 2
    ShortCut = 16467
    OnExecute = actnSaveExecute
  end
  <------->Object actnExit: TAction
    Category = 'File'
    Caption = 'E&xit'
    OnExecute = actnExitExecute
  end
 end
 <------->Object ImageList: TImageList
  Left = 304
  Top = 40
  Bitmap = {
 end
End

\\\\
<------->Object AboutForm: TAboutForm
 Left = 164
 Top = 119
 Width = 365
 Height = 220
 VertScrollBar.Range = 193
 HorzScrollBar.Range = 353
 ActiveControl = Button1
 BorderStyle = fbsDialog
 Caption = 'About'
 Color = clBackground
 Constraints.MaxHeight = 220
 Constraints.MaxWidth = 365
 Constraints.MinHeight = 220
 Constraints.MinWidth = 365
```

```
    Category = 'Edit'
    Caption = '&Copy'
    Hint = 'Copy'
    ImageIndex = 4
    ShortCut = 16451
  end
  <------->Object actnOpen: TAction
    Category = 'File'
    Caption = '&Open'
    Hint = 'Open a text file'
    ImageIndex = 1
    OnExecute = actnOpenExecute
  end
  <------->Object actnPaste: TEditPaste
    Category = 'Edit'
    Caption = '&Paste'
    Hint = 'Paste'
    ImageIndex = 5
    ShortCut = 16470
  end
  <------->Object actnSelectAll: TEditSelectAll
    Category = 'Edit'
    Caption = 'Select &All'
  end
  <------->Object actnDelete: TEditDelete
    Category = 'Edit'
    Caption = '&Delete'
    ImageIndex = 6
    ShortCut = 4103
  end
  <------->Object actnWinClose: TWindowClose
    Category = 'Window'
    Caption = 'C&lose'
    OnExecute = actnWinCloseExecute
  end
  <------->Object actnWinCascade: TWindowCascade
    Category = 'Window'
    Caption = '&Cascade'
  end
  <------->Object actnWinTile: TWindowTile
```

```
  TabOrder = 0
 end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

Procedures29

الملف التنفيذي لبرنامج

(docking)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit uMain;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  Menus, StdCtrls, ComCtrls,
ActnList, ToolWin, ExtCtrls,
uDockForm;

type
  TMainForm = class(TForm)
    CoolBar1: TCoolBar;
    ToolBar1: TToolBar;
    ToolBar2: TToolBar;
    ToolButton1: TToolButton;
    ToolButton2: TToolButton;
    ToolButton3: TToolButton;
    ToolButton5: TToolButton;
    ToolButton6: TToolButton;
    ToolButton7: TToolButton;
    ToolButton13: TToolButton;
    btnToolBar1: TToolButton;
    btnToolBar2: TToolButton;
    ActionList1: TActionList;
    ViewToolBar1: TAction;
    ViewToolBar2: TAction;
    LeftDockPanel: TPanel;
    BottomDockPanel: TPanel;
```

```
  Scaled = False
  PixelsPerInch = 96
  <------->Object Button1: TButton
    Left = 144
    Top = 168
    Width = 75
    Height = 25
    Caption = 'OK'
    TabOrder = 1
    OnClick = Button1Click
  end
  <------->Object Label1: TLabel
    Left = 16
    Top = 16
    Width = 337
    Height = 30
    AutoSize = False
    Caption = 'Simple Text Editor (c)
2001 Borland Software Corporation'
    Layout = tlCenter
  end
End



\\\\\\

<------->Object EditForm:
TEditForm
  Left = 162
  Top = 119
  Width = 357
  Height = 267
  ActiveControl = memText
  Caption = 'EditForm'
  Color = clBackground
  FormStyle = fsMDIChild
  Position = poDefault
  OnClose = FormClose
  PixelsPerInch = 75
  TextHeight = 13
  TextWidth = 6
  <------->Object memText: TMemo
    Left = 0
    Top = 0
    Width = 357
    Height = 267
    Align = alClient
```

```
TObject;
    Source: TDragDockObject; X,
Y: Integer; State: TDragState;
    var Accept: Boolean);
<-------> procedure
BottomDockPanelDockOver(Sender
: TObject;
    Source: TDragDockObject; X,
Y: Integer; State: TDragState;
    var Accept: Boolean);
<-------> procedure
LeftDockPanelUnDock(Sender:
TObject; Client: TControl;
    NewTarget: TWinControl; var
Allow: Boolean);
<-------> procedure
ExitActionExecute(Sender:
TObject);
<-------> procedure
ViewWhiteWindowExecute(Sender:
TObject);
<-------> procedure
LeftDockPanelGetSiteInfo(Sender:
TObject;
    DockClient: TControl; var
InfluenceRect: TRect; MousePos:
TPoint;
    var CanDock: Boolean);
  private
<-------> procedure
CreateDockableWindows;
  public
<-------> procedure
ShowDockPanel(APanel: TPanel;
MakeVisible: Boolean; Client:
TControl);
  end;

var
  MainForm: TMainForm;

implementation

uses uTabHost, uConjoinHost;

{$R *.dfm}

const
  Colors: array[0..6] of TColor =
```

```
    VSplitter: TSplitter;
    HSplitter: TSplitter;
    MainMenu1: TMainMenu;
    File2: TMenuItem;
    Exit2: TMenuItem;
    View2: TMenuItem;
    N2: TMenuItem;
    ToolBar21: TMenuItem;
    ToolBar11: TMenuItem;
    ToolButton16: TToolButton;
    ViewWhiteWindow: TAction;
    ExitAction: TAction;
    ViewBlueWindow: TAction;
    ViewGreenWindow: TAction;
    ViewRedWindow: TAction;
    ViewTealWindow: TAction;
    ViewPurpleWindow: TAction;
    ViewLimeWindow: TAction;
    ToolButton4: TToolButton;
    White1: TMenuItem;
    Blue1: TMenuItem;
    Green1: TMenuItem;
    Lime1: TMenuItem;
    Purple1: TMenuItem;
    Red1: TMenuItem;
    Teal1: TMenuItem;
<-------> procedure
FormCreate(Sender: TObject);
<-------> procedure
CoolBar1DockOver(Sender:
TObject; Source:
TDragDockObject; X,
     Y: Integer; State: TDragState;
var Accept: Boolean);
<-------> procedure
ViewToolBar1Execute(Sender:
TObject);
<-------> procedure
ViewToolBar2Execute(Sender:
TObject);
<-------> procedure
Exit1Click(Sender: TObject);
<-------> procedure
LeftDockPanelDockDrop(Sender:
TObject;
     Source: TDragDockObject; X,
Y: Integer);
<-------> procedure
LeftDockPanelDockOver(Sender:
```

```
nder: TObject; Source: TDragDockObject;
  X, Y: Integer; State: TDragState; var Accept: Boolean);
var
  ARect: TRect;
begin
  Accept := (Source.Control is TToolBar);
  if Accept then
  begin
    //Modify the DockRect to preview dock area (Coolbar client area)
    ARect.TopLeft := CoolBar1.ClientToScreen(CoolBar1.ClientRect.TopLeft);
    ARect.BottomRight := CoolBar1.ClientToScreen(CoolBar1.ClientRect.BottomRight);
    Source.DockRect := ARect;
  end;
end;

procedure TMainForm.ViewToolBar1Execute(Sender: TObject);
begin
  //Toggles the visible state of Toolbar1, regardless of it's docked state.
  ToolBar11.Checked := not ToolBar11.Checked;
  btnToolBar1.Down := ToolBar11.Checked;
  if ToolBar1.Floating then
    ToolBar1.HostDockSite.Visible := ToolBar11.Checked
  else
    ToolBar1.Visible := ToolBar11.Checked;
end;

procedure TMainForm.ViewToolBar2Execute(Sender: TObject);
begin
  //Toggles the visible state of Toolbar2, regardless of it's docked state.
```

```
(clWhite, clBlue, clGreen, clRed, clTeal,
                                   clPurple, clLime);
  ColStr: array[0..6] of string = ('White', 'Blue', 'Green', 'Red', 'Teal',
                                   'Purple', 'Lime');

var
  DockWindows: array[0..6] of TDockableForm;

{TMainForm}

procedure TMainForm.FormCreate(Sender: TObject);
begin
  CreateDockableWindows;
end;

procedure TMainForm.CreateDockableWindows;
var
  I: Integer;
begin
  for I := 0 to High(DockWindows) do
  begin
    DockWindows[I] := TDockableForm.Create(Application);
    DockWindows[I].Caption := ColStr[I];
    DockWindows[I].Memo1.Color := Colors[I];
    DockWindows[I].Memo1.Font.Color := Colors[I] xor $00FFFFFF;
    DockWindows[I].Memo1.Text := ColStr[I] + ' window ';
  end;
end;

procedure TMainForm.CoolBar1DockOver(Se
```

```
  ARect: TRect;
begin
  Accept := Source.Control is TDockableForm;
  if Accept then
  begin
    //Modify the DockRect to preview dock area.
    ARect.TopLeft := LeftDockPanel.ClientToScreen(Point(0, 0));
    ARect.BottomRight := LeftDockPanel.ClientToScreen(
      Point(Self.ClientWidth div 3, LeftDockPanel.Height));
    Source.DockRect := ARect;
  end;
end;

procedure TMainForm.BottomDockPanelDockOver(Sender: TObject;
  Source: TDragDockObject; X, Y: Integer; State: TDragState;
  var Accept: Boolean);
var
  ARect: TRect;
begin
  Accept := Source.Control is TDockableForm;
  if Accept then
  begin
    //Modify the DockRect to preview dock area.
    ARect.TopLeft := BottomDockPanel.ClientToScreen(
      Point(0, -Self.ClientHeight div 3));
    ARect.BottomRight := BottomDockPanel.ClientToScreen(
      Point(BottomDockPanel.Width, BottomDockPanel.Height));
    Source.DockRect := ARect;
  end;
end;

procedure TMainForm.LeftDockPanelUnDock(Sender: TObject; Client: TControl;
```

```
  ToolBar21.Checked := not ToolBar21.Checked;
  btnToolBar2.Down := ToolBar21.Checked;
  if ToolBar2.Floating then
    TToolDockForm(ToolBar2.HostDockSite).Visible := ToolBar21.Checked
  else
    ToolBar2.Visible := ToolBar21.Checked;
end;

procedure TMainForm.Exit1Click(Sender: TObject);
begin
  Close;
end;

procedure TMainForm.LeftDockPanelDockDrop(Sender: TObject;
  Source: TDragDockObject; X, Y: Integer);
begin
  //OnDockDrop gets called AFTER the client has actually docked,
  //so we check for DockClientCount = 1 before making the dock panel visible.
  if (Sender as TPanel).DockClientCount = 1 then
    ShowDockPanel(Sender as TPanel, True, nil);
  (Sender as TPanel).DockManager.ResetBounds(True);
  //Make DockManager repaints it's clients.
end;

procedure TMainForm.LeftDockPanelDockOver(Sender: TObject;
  Source: TDragDockObject; X, Y: Integer; State: TDragState;
  var Accept: Boolean);
var
```

```
    if APanel = LeftDockPanel then
    begin
      APanel.Width := ClientWidth
div 3;
      VSplitter.Left := APanel.Width
+ VSplitter.Width;
    end
    else begin
      APanel.Height := ClientHeight
div 3;
      HSplitter.Top := ClientHeight -
APanel.Height - HSplitter.Width;
    end
  else
    if APanel = LeftDockPanel then
      APanel.Width := 0
    else
      APanel.Height := 0;
  if MakeVisible and (Client <> nil)
then Client.Show;
end;

procedure
TMainForm.ExitActionExecute(Sen
der: TObject);
begin
  Close;
end;

procedure
TMainForm.ViewWhiteWindowExe
cute(Sender: TObject);
var
  DockWindow: TDockableForm;
begin
  DockWindow :=
DockWindows[(Sender as
TComponent).Tag];
  with DockWindow do
   //if the docked window is
TabDocked, it is docked to the
PageControl
   //(owned by TTabDockHost) so
show the host form.
   if HostDockSite is TPageControl
then

TTabDockHost(HostDockSite.Owne
r).Show
```

```
  NewTarget: TWinControl; var
Allow: Boolean);
begin
  //OnUnDock gets called BEFORE
the client is undocked, in order to
optionally
  //disallow the undock.
DockClientCount is never 0 when
called from this event.
  if (Sender as
TPanel).DockClientCount = 1 then
    ShowDockPanel(Sender as
TPanel, False, nil);
end;

procedure
TMainForm.ShowDockPanel(APane
l: TPanel; MakeVisible: Boolean;
Client: TControl);
begin
  //Client - the docked client to show
if we are re-showing the panel.
  //Client is ignored if hiding the
panel.

  //Since docking to a non-visible
docksite isn't allowed, instead of
setting
  //Visible for the panels we set the
width to zero. The default
InfluenceRect
  //for a control extends a few pixels
beyond it's boundaries, so it is
possible
  //to dock to zero width controls.

  //Don't try to hide a panel which
has visible dock clients.
  if not MakeVisible and
(APanel.VisibleDockClientCount >
1) then
    Exit;

  if APanel = LeftDockPanel then
    VSplitter.Visible := MakeVisible
  else
    HSplitter.Visible := MakeVisible;

    if MakeVisible then
```

```
unit uConjoinHost;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, uDockForm;

type
  TConjoinDockHost = class(TForm)
<------->    procedure FormClose(Sender: TObject; var Action: TCloseAction);
<------->    procedure FormDockDrop(Sender: TObject; Source: TDragDockObject; X,
      Y: Integer);
<------->    procedure FormUnDock(Sender: TObject; Client: TControl;
      NewTarget: TWinControl; var Allow: Boolean);
<------->    procedure FormDockOver(Sender: TObject; Source: TDragDockObject; X,
      Y: Integer; State: TDragState; var Accept: Boolean);
<------->    procedure FormGetSiteInfo(Sender: TObject; DockClient: TControl;
      var InfluenceRect: TRect; MousePos: TPoint; var CanDock: Boolean);
  private
<------->    procedure DoFloat(AControl: TControl);
  public
<------->    procedure UpdateCaption(Exclude: TControl);
  end;

var
  ConjoinDockHost: TConjoinDockHost;

implementation

{$R *.dfm}
```

```
   else
     //If window is conjoin-docked, host and/or form may not be visible
     //so show both.
     if (HostDockSite is TConjoinDockHost) and not HostDockSite.Visible then
     begin
       HostDockSite.Show;
       TConjoinDockHost(HostDockSite).UpdateCaption(nil);
       DockWindow.Show;
     end else
       //If form is docked to one of the "hidden" docking panels, resize the
       //panel and re-show the docked form.
       if (HostDockSite is TPanel)
       and ((HostDockSite.Height = 0) or (HostDockSite.Width = 0)) then
         MainForm.ShowDockPanel(HostDockSite as TPanel, True, DockWindow)
       else
         //if the window isn't docked at all, simply show it.
         DockWindow.Show;
end;

procedure TMainForm.LeftDockPanelGetSiteInfo(Sender: TObject;
  DockClient: TControl; var InfluenceRect: TRect; MousePos: TPoint;
  var CanDock: Boolean);
begin
  //if CanDock is true, the panel will not automatically draw the preview rect.
  CanDock := DockClient is TDockableForm;
end;

end.

\\\\\\\\\\\\\\\\\\
```

```
    Caption := Caption +
TDockableForm(DockClients[I]).Caption + ' ';
end;

procedure TConjoinDockHost.FormDockDrop(Sender: TObject;
  Source: TDragDockObject; X, Y: Integer);
begin
  UpdateCaption(nil);
  DockManager.ResetBounds(True);
  //Force DockManager to redraw it's clients.
end;

procedure TConjoinDockHost.FormUnDock(Sender: TObject; Client: TControl;
  NewTarget: TWinControl; var Allow: Boolean);
begin
  //only 2 dock clients means the host must be destroyed and
  //the remaining window undocked to its old position and size.
  //(Recall that OnUnDock gets called before the undocking actually occurs)
  if Client is TDockableForm then
    TDockableForm(Client).DockSite := True;
  if (DockClientCount = 2) and (NewTarget <> Self) then
    PostMessage(Self.Handle, WM_CLOSE, 0, 0);
  UpdateCaption(Client);
end;

procedure TConjoinDockHost.FormDockOver(Sender: TObject;
  Source: TDragDockObject; X, Y: Integer; State: TDragState;
  var Accept: Boolean);
begin
  Accept := Source.Control is TDockableForm;
```

```

procedure TConjoinDockHost.DoFloat(AControl: TControl);
var
  ARect: TRect;
begin
  //float the control with its original size.
  ARect.TopLeft := AControl.ClientToScreen(Point(0, 0));
  ARect.BottomRight := AControl.ClientToScreen(Point(AControl.UndockWidth,

AControl.UndockHeight));
  AControl.ManualFloat(ARect);
end;

procedure TConjoinDockHost.FormClose(Sender: TObject;
  var Action: TCloseAction);
begin
  if DockClientCount = 1 then
  begin
    DoFloat(DockClients[0]);
    Action := caFree;
  end else
    Action := caHide;
end;

procedure TConjoinDockHost.UpdateCaption(Exclude: TControl);
var
  I: Integer;
begin
  //if a dockable form is undocking, it will pass itself in as Exclude
  //because even it hasn't actually been taken out of the DockClient array
  //at this point.
  Caption := '';
  for I := 0 to DockClientCount-1 do
    if DockClients[I].Visible and (DockClients[I] <> Exclude) then
```

```
CM_DOCKCLIENT;
  public
  end;


implementation

{$R *.dfm}

uses ComCtrls, uTabHost,
uConjoinHost, uMain;

procedure
TDockableForm.FormDockOver(Se
nder: TObject;
  Source: TDragDockObject; X, Y:
Integer; State: TDragState;
  var Accept: Boolean);
var
  ARect: TRect;
begin
  Accept := (Source.Control is
TDockableForm);
  //Draw dock preview depending on
where the cursor is relative to our
client area
  if Accept and
(ComputeDockingRect(ARect,
Point(X, Y)) <> alNone) then
    Source.DockRect := ARect;
end;

function
TDockableForm.ComputeDockingR
ect(var DockRect: TRect;
MousePos: TPoint): TAlign;
var
  DockTopRect,
  DockLeftRect,
  DockBottomRect,
  DockRightRect,
  DockCenterRect: TRect;
begin
  Result := alNone;
  //divide form up into docking
"Zones"
  DockLeftRect.TopLeft := Point(0,
0);
  DockLeftRect.BottomRight :=
```

```
end;

procedure
TConjoinDockHost.FormGetSiteInf
o(Sender: TObject;
  DockClient: TControl; var
InfluenceRect: TRect; MousePos:
TPoint;
  var CanDock: Boolean);
begin
  CanDock := DockClient is
TDockableForm;
end;

end.

\\\\\\\\\\\\\\\\

unit uDockForm;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs, Menus,
  ExtCtrls, StdCtrls;

type
  TDockableForm = class(TForm)
    Memo1: TMemo;
<-------> procedure
FormDockOver(Sender: TObject;
Source: TDragDockObject; X,
      Y: Integer; State: TDragState;
var Accept: Boolean);
<-------> procedure
FormClose(Sender: TObject; var
Action: TCloseAction);
<-------> procedure
FormShow(Sender: TObject);
  private
    function
ComputeDockingRect(var
DockRect: TRect; MousePos:
TPoint): TAlign;
<-------> procedure
CMDockClient(var Message:
TCMDockClient); message
```

```
ClientHeight div 2;
    end
   else
    if PtInRect(DockRightRect,
MousePos) then
      begin
        Result := alRight;
        DockRect := DockRightRect;
        DockRect.Left := ClientWidth
div 2;
      end
    else
      if PtInRect(DockBottomRect,
MousePos) then
        begin
          Result := alBottom;
          DockRect :=
DockBottomRect;
          DockRect.Left := 0;
          DockRect.Right :=
ClientWidth;
          DockRect.Top :=
ClientHeight div 2;
        end
      else
        if PtInRect(DockCenterRect,
MousePos) then
        begin
          Result := alClient;
          DockRect :=
DockCenterRect;
        end;
  if Result = alNone then Exit;

  //DockRect is in screen coordinates.
  DockRect.TopLeft :=
ClientToScreen(DockRect.TopLeft);
  DockRect.BottomRight :=
ClientToScreen(DockRect.BottomRi
ght);
end;

procedure
TDockableForm.FormClose(Sender:
TObject;
  var Action: TCloseAction);
begin
  //the action taken depends on how
the form is docked.
```

```
Point(ClientWidth div 5,
ClientHeight);

  DockTopRect.TopLeft :=
Point(ClientWidth div 5, 0);
  DockTopRect.BottomRight :=
Point(ClientWidth div 5 * 4,
ClientHeight div 5);

  DockRightRect.TopLeft :=
Point(ClientWidth div 5 * 4, 0);
  DockRightRect.BottomRight :=
Point(ClientWidth, ClientHeight);

  DockBottomRect.TopLeft :=
Point(ClientWidth div 5,
ClientHeight div 5 * 4);
  DockBottomRect.BottomRight :=
Point(ClientWidth div 5 * 4,
ClientHeight);

  DockCenterRect.TopLeft :=
Point(ClientWidth div 5,
ClientHeight div 5);
  DockCenterRect.BottomRight :=
Point(ClientWidth div 5 * 4,
ClientHeight div 5 * 4);

  //Find out where the mouse cursor
is, to decide where to draw dock
preview.
  if PtInRect(DockLeftRect,
MousePos) then
    begin
      Result := alLeft;
      DockRect := DockLeftRect;
      DockRect.Right := ClientWidth
div 2;
    end
  else
    if PtInRect(DockTopRect,
MousePos) then
      begin
        Result := alTop;
        DockRect := DockTopRect;
        DockRect.Left := 0;
        DockRect.Right :=
ClientWidth;
        DockRect.Bottom :=
```

```
ManualDock can be safely called during a drag
  //operation is we override processing of CM_DOCKCLIENT.

  if Message.DockSource.Control is TDockableForm then
  begin

    //Find out how to dock (Using a TAlign as the result of ComputeDockingRect)
    Pt.x := Message.MousePos.x;
    Pt.y := Message.MousePos.y;
    DockType := ComputeDockingRect(ARect, Pt);

    //if we are over a dockable form docked to a panel in the
    //main window, manually dock the dragged form to the panel with
    //the correct orientation.
    if (HostDockSite is TPanel) then
    begin
      Message.DockSource.Control.ManualDock(HostDockSite, nil, DockType);
      Exit;
    end;

    //alClient => Create a TabDockHost and manually dock both forms to the PageControl
    //owned by the TabDockHost.
    if DockType = alClient then
    begin
      Host := TTabDockHost.Create(Application);
      Host.BoundsRect := Self.BoundsRect;
      Self.ManualDock(TTabDockHost(Host).PageControl1, nil, alClient);
      Message.DockSource.Control.ManualDock(TTabDockHost(Host).PageControl1, nil, alClient);
      Host.Visible := True;

  if (HostDockSite is TConjoinDockHost) then
  begin
    //remove the form's caption from the conjoin dock host's caption list
    TConjoinDockHost(HostDockSite).UpdateCaption(Self);

    //if we're the last visible form on a conjoined form, hide the form
    if HostDockSite.VisibleDockClientCount <= 1 then
      HostDockSite.Hide;
  end;

  //if docked to a panel, tell the panel to hide itself. If there are other
  //visible dock clients on the panel, it ShowDockPanel won't allow it to
  //be hidden
  if (HostDockSite is TPanel) then
    MainForm.ShowDockPanel(HostDockSite as TPanel, False, nil);

  Action := caHide;
end;

procedure TDockableForm.CMDockClient(var Message: TCMDockClient);
var
  ARect: TRect;
  DockType: TAlign;
  Host: TForm;
  Pt: TPoint;
begin
  //Overriding this message allows the dock form to create host forms
  //depending on the mouse position when docking occurs. If we don't override
  //this message, the form will use VCL's default DockManager.

  //NOTE: the only time
```

```
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ComCtrls;

type
  TTabDockHost = class(TForm)
    PageControl1: TPageControl;
<------->    procedure FormClose(Sender: TObject; var Action: TCloseAction);
<------->    procedure PageControl1UnDock(Sender: TObject; Client: TControl;
      NewTarget: TWinControl; var Allow: Boolean);
<------->    procedure PageControl1GetSiteInfo(Sender: TObject;
      DockClient: TControl; var InfluenceRect: TRect; MousePos: TPoint;
      var CanDock: Boolean);
<------->    procedure PageControl1DockOver(Sender: TObject;
      Source: TDragDockObject; X, Y: Integer; State: TDragState;
      var Accept: Boolean);
  private

   { Private declarations }
  public
    { Public declarations }
  end;

var
  TabDockHost: TTabDockHost;

implementation

{$R *.dfm}

uses uDockForm;

procedure TTabDockHost.FormClose(Sender: TObject;
```

```
    end
    //if DockType <> alClient, create the ConjoinDockHost and manually dock both
    //forms to it. Be sure to make dockable forms non-dockable when hosted by
    // ConjoinDockForm, since it is using the VCL default DockManager.
    else begin
      Host := TConjoinDockHost.Create(Application);
      Host.BoundsRect := Self.BoundsRect;
      Self.ManualDock(Host, nil, alNone);
      Self.DockSite := False;
      Message.DockSource.Control.ManualDock(Host, nil, DockType);
      TDockableForm(Message.DockSource.Control).DockSite := False;
      Host.Visible := True;
    end;
  end;
end;

procedure TDockableForm.FormShow(Sender: TObject);
begin
  if HostDockSite is TConjoinDockHost then
    TConjoinDockHost(HostDockSite).UpdateCaption(nil);
end;

end.
\\\\\\\\\\


unit uTabHost;

interface
```

```
  var Action: TCloseAction);
var
  ARect: TRect;
begin
  if PageControl1.DockClientCount = 1 then
  begin
    with PageControl1.DockClients[0] do
    begin
      ARect.TopLeft := ClientToScreen(Point(0, 0));
      ARect.BottomRight := ClientToScreen(Point(UndockWidth, UndockHeight));
      ManualFloat(ARect);
    end;
    Action := caFree;
  end else
    Action := caHide;
end;

procedure TTabDockHost.PageControl1UnDock(Sender: TObject;
  Client: TControl; NewTarget: TWinControl; var Allow: Boolean);
begin
  //only 2 dock clients means the host must be destroyed and
  //the remaining window undocked to its old position and size.
  if (PageControl1.DockClientCount = 2) and (NewTarget <> Self) then
    PostMessage(Self.Handle, WM_CLOSE, 0, 0);
end;

procedure TTabDockHost.PageControl1GetSiteInfo(Sender: TObject;
  DockClient: TControl; var InfluenceRect: TRect; MousePos: TPoint;
  var CanDock: Boolean);
begin
  CanDock := DockClient is TDockableForm;
end;

procedure TTabDockHost.PageControl1DockOver(Sender: TObject;
  Source: TDragDockObject; X, Y: Integer; State: TDragState;
  var Accept: Boolean);
begin
  Accept := Source.Control is TDockableForm;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

<-------> Object s94

الملفالنصي لبرنامج

(Docking)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<-------> Object MainForm: TMainForm
  Left = -4
  Top = -4
  Width = 1032
  Height = 746
  Caption = 'Docking Demo'
  Color = clWindow
  ParentFont = True
  Menu = MainMenu1
  OldCreateOrder = False
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> Object VSplitter: TSplitter
    Left = 0
    Top = 52
    Width = 4
    Height = 636
```

Caption = 'ToolBar1'
Constraints.MaxWidth = 280
DragKind = dkDock
DragMode = dmAutomatic
Flat = True
ShowCaptions = True
TabOrder = 0
Transparent = True
Wrapable = False
<-------> Object ToolButton13: TToolButton
Left = 0
Top = 0
Action = ExitAction
end
<-------> Object ToolButton16: TToolButton
Left = 59
Top = 0
Width = 14
Caption = 'ToolButton16'
ImageIndex = 7
Style = tbsSeparator
end
<-------> Object btnToolBar1: TToolButton
Left = 73
Top = 0
Action = ViewToolBar1
Style = tbsCheck
end
<-------> Object btnToolBar2: TToolButton
Left = 132
Top = 0
Action = ViewToolBar2
Style = tbsCheck
end
end
<-------> Object ToolBar2: TToolBar
Left = 9
Top = 25
Width = 390
Height = 23
AutoSize = True
ButtonHeight = 21
ButtonWidth = 43

Visible = False
end
<-------> Object HSplitter: TSplitter
Left = 0
Top = 688
Width = 1024
Height = 4
Cursor = crVSplit
Align = alBottom
Visible = False
end
<-------> Object CoolBar1: TCoolBar
Left = 0
Top = 0
Width = 1024
Height = 52
AutoSize = True
BandMaximize = bmDblClick
Bands = <
item
Break = False
Control = ToolBar1
ImageIndex = -1
MinHeight = 23
Width = 1020
end
item
Control = ToolBar2
ImageIndex = -1
MinHeight = 23
Width = 1020
end>
Color = clMenu
DockSite = True
ParentColor = False
OnDockOver = CoolBar1DockOver
<-------> Object ToolBar1: TToolBar
Left = 9
Top = 0
Width = 280
Height = 23
AutoSize = True
ButtonHeight = 21
ButtonWidth = 59

```
      Top = 0
      Action = ViewTealWindow
    end
  end
 end
 <-------> Object  LeftDockPanel: TPanel
  Left = 4
  Top = 52
  Width = 0
  Height = 636
  Align = alLeft
  DockSite = True
  TabOrder = 1
  OnDockDrop = LeftDockPanelDockDrop
  OnDockOver = LeftDockPanelDockOver
  OnGetSiteInfo = LeftDockPanelGetSiteInfo
  OnUnDock = LeftDockPanelUnDock
 end
 <-------> Object  BottomDockPanel: TPanel
  Left = 0
  Top = 688
  Width = 1024
  Height = 0
  Align = alBottom
  DockSite = True
  TabOrder = 2
  OnDockDrop = LeftDockPanelDockDrop
  OnDockOver = BottomDockPanelDockOver
  OnGetSiteInfo = LeftDockPanelGetSiteInfo
  OnUnDock = LeftDockPanelUnDock
 end
 <-------> Object  ActionList1: TActionList
  Left = 136
  Top = 80
  <-------> Object  ViewToolBar1: TAction
   Category = 'ViewToolBars'
   Caption = 'ToolBar &1'
    Caption = 'ToolBar2'
    Constraints.MaxWidth = 390
    DragKind = dkDock
    DragMode = dmAutomatic
    Flat = True
    ShowCaptions = True
    TabOrder = 1
    Transparent = True
    Wrapable = False
    <-------> Object  ToolButton1: TToolButton
     Left = 0
     Top = 0
     Action = ViewWhiteWindow
    end
    <-------> Object  ToolButton2: TToolButton
     Left = 43
     Top = 0
     Action = ViewBlueWindow
    end
    <-------> Object  ToolButton3: TToolButton
     Left = 86
     Top = 0
     Action = ViewGreenWindow
    end
    <-------> Object  ToolButton5: TToolButton
     Left = 129
     Top = 0
     Action = ViewLimeWindow
    end
    <-------> Object  ToolButton6: TToolButton
     Left = 172
     Top = 0
     Action = ViewPurpleWindow
    end
    <-------> Object  ToolButton7: TToolButton
     Left = 215
     Top = 0
     Action = ViewRedWindow
    end
    <-------> Object  ToolButton4: TToolButton
     Left = 258
```

```
ViewWhiteWindowExecute
  end
  <-------> Object
ViewTealWindow: TAction
    Tag = 4
    Category = 'ViewWindows'
    Caption = '&Teal'
    OnExecute =
ViewWhiteWindowExecute
  end
  <-------> Object
ViewPurpleWindow: TAction
    Tag = 5
    Category = 'ViewWindows'
    Caption = '&Purple'
    OnExecute =
ViewWhiteWindowExecute
  end
  <-------> Object
ViewLimeWindow: TAction
    Tag = 6
    Category = 'ViewWindows'
    Caption = '&Lime'
    OnExecute =
ViewWhiteWindowExecute
  end
 end
 <-------> Object  MainMenu1:
TMainMenu
  Left = 176
  Top = 80
  <-------> Object  File2:
TMenuItem
    Caption = '&File'
    <-------> Object  Exit2:
TMenuItem
      Action = ExitAction
    end
  end
  <-------> Object  View2:
TMenuItem
    Caption = '&View'
    <-------> Object  ToolBar11:
TMenuItem
      Action = ViewToolBar1
    end
    <-------> Object  ToolBar21:
TMenuItem
      Action = ViewToolBar2
```

```
    Checked = True
    ImageIndex = 1
    OnExecute =
ViewToolBar1Execute
  end
  <-------> Object  ViewToolBar2:
TAction
    Category = 'ViewToolBars'
    Caption = 'ToolBar &2'
    Checked = True
    ImageIndex = 2
    OnExecute =
ViewToolBar2Execute
  end
  <-------> Object
ViewWhiteWindow: TAction
    Category = 'ViewWindows'
    Caption = '&White'
    Hint = 'View white window'
    OnExecute =
ViewWhiteWindowExecute
  end
  <-------> Object  ExitAction:
TAction
    Caption = 'E&xit'
    OnExecute = ExitActionExecute
  end
  <-------> Object
ViewBlueWindow: TAction
    Tag = 1
    Category = 'ViewWindows'
    Caption = '&Blue'
    OnExecute =
ViewWhiteWindowExecute
  end
  <-------> Object
ViewGreenWindow: TAction
    Tag = 2
    Category = 'ViewWindows'
    Caption = '&Green'
    OnExecute =
ViewWhiteWindowExecute
  end
  <-------> Object
ViewRedWindow: TAction
    Tag = 3
    Category = 'ViewWindows'
    Caption = '&Red'
    OnExecute =
```

```
  Font.Height = -10
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  OnClose = FormClose
  PixelsPerInch = 96
  TextHeight = 13
  <--------> Object  PageControl1:
TPageControl
    Left = 0
    Top = 0
    Width = 453
    Height = 243
    Align = alClient
    DockSite = True
    TabOrder = 0
    OnDockOver =
PageControl1DockOver
    OnGetSiteInfo =
PageControl1GetSiteInfo
    OnUnDock =
PageControl1UnDock
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
<--------> Object  ConjoinDockHost:
TConjoinDockHost
  Left = 152
  Top = 118
  Width = 606
  Height = 337
  BorderStyle = bsSizeToolWin
  Caption = 'ConjoinDockHost'
  Color = clBtnFace
  UseDockManager = True
  DockSite = True
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -10
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  OnClose = FormClose
  OnDockDrop = FormDockDrop
  OnDockOver = FormDockOver
  OnGetSiteInfo = FormGetSiteInfo
  OnUnDock = FormUnDock
```

```
      end
      <--------> Object  N2: TMenuItem
        Caption = '-'
      end
      <--------> Object  White1:
TMenuItem
        Action = ViewWhiteWindow
      end
      <--------> Object  Blue1:
TMenuItem
        Action = ViewBlueWindow
      end
      <--------> Object  Green1:
TMenuItem
        Action = ViewGreenWindow
      end
      <--------> Object  Lime1:
TMenuItem
        Action = ViewLimeWindow
      end
      <--------> Object  Purple1:
TMenuItem
        Action = ViewPurpleWindow
      end
      <--------> Object  Red1:
TMenuItem
        Action = ViewRedWindow
      end
      <--------> Object  Teal1:
TMenuItem
        Action = ViewTealWindow
      end
    end
  end
End
\\\

<--------> Object  TabDockHost:
TTabDockHost
  Left = 184
  Top = 119
  Width = 461
  Height = 277
  Caption = 'TabDockHost'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
```

Caption = 'Docking Demo'
Color = clWindow
ParentFont = True
Menu = MainMenu1
OldCreateOrder = False
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> Object VSplitter: TSplitter
Left = 0
Top = 52
Width = 4
Height = 388
Visible = False
end
<-------> Object HSplitter: TSplitter
Left = 0
Top = 440
Width = 675
Height = 4
Cursor = crVSplit
Align = alBottom
Visible = False
end
<-------> Object CoolBar1: TCoolBar
Left = 0
Top = 0
Width = 675
Height = 52
AutoSize = True
BandMaximize = bmDblClick
Bands = <
item
Break = False
Control = ToolBar1
ImageIndex = -1
MinHeight = 23
Width = 671
end
item
Control = ToolBar2
ImageIndex = -1
MinHeight = 23
Width = 671
end>

PixelsPerInch = 96
TextHeight = 13
End
\\
<-------> Object DockableForm: TDockableForm
Left = 138
Top = 115
Width = 397
Height = 241
Caption = 'DockableForm'
Color = clBtnFace
DockSite = True
DragKind = dkDock
DragMode = dmAutomatic
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -10
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = False
OnClose = FormClose
OnDockOver = FormDockOver
OnShow = FormShow
PixelsPerInch = 96
TextHeight = 13
<-------> Object Memo1: TMemo
Left = 0
Top = 0
Width = 389
Height = 207
Align = alClient
Lines.Strings = (
'Memo1')
TabOrder = 0
end
End
\\
<-------> Object MainForm: TMainForm
Left = 181
Top = 152
Width = 683
Height = 498

```
    Color = clMenu
    DockSite = True
    ParentColor = False
    OnDockOver =
CoolBar1DockOver
    <-------> Object ToolBar1:
TToolBar
      Left = 9
      Top = 0
      Width = 280
      Height = 23
      AutoSize = True
      ButtonHeight = 21
      ButtonWidth = 59
      Caption = 'ToolBar1'
      Constraints.MaxWidth = 280
      DragKind = dkDock
      DragMode = dmAutomatic
      Flat = True
      ShowCaptions = True
      TabOrder = 0
      Transparent = True
      Wrapable = False
      <-------> Object ToolButton13:
TToolButton
        Left = 0
        Top = 0
        Action = ExitAction
      end
      <-------> Object ToolButton16:
TToolButton
        Left = 59
        Top = 0
        Width = 14
        Caption = 'ToolButton16'
        ImageIndex = 7
        Style = tbsSeparator
      end
      <-------> Object btnToolBar1:
TToolButton
        Left = 73
        Top = 0
        Action = ViewToolBar1
        Style = tbsCheck
      end
      <-------> Object btnToolBar2:
TToolButton
        Left = 132
        Top = 0
        Action = ViewToolBar2
        Style = tbsCheck
      end
    end
    <-------> Object ToolBar2:
TToolBar
      Left = 9
      Top = 25
      Width = 390
      Height = 23
      AutoSize = True
      ButtonHeight = 21
      ButtonWidth = 43
      Caption = 'ToolBar2'
      Constraints.MaxWidth = 390
      DragKind = dkDock
      DragMode = dmAutomatic
      Flat = True
      ShowCaptions = True
      TabOrder = 1
      Transparent = True
      Wrapable = False
      <-------> Object ToolButton1:
TToolButton
        Left = 0
        Top = 0
        Action = ViewWhiteWindow
      end
      <-------> Object ToolButton2:
TToolButton
        Left = 43
        Top = 0
        Action = ViewBlueWindow
      end
      <-------> Object ToolButton3:
TToolButton
        Left = 86
        Top = 0
        Action = ViewGreenWindow
      end
      <-------> Object ToolButton5:
TToolButton
        Left = 129
        Top = 0
        Action = ViewLimeWindow
      end
      <-------> Object ToolButton6:
```

```
   OnDockOver =
BottomDockPanelDockOver
   OnGetSiteInfo =
LeftDockPanelGetSiteInfo
   OnUnDock =
LeftDockPanelUnDock
  end
  <-------> Object  ActionList1:
TActionList
   Left = 136
   Top = 80
   <-------> Object  ViewToolBar1:
TAction
     Category = 'ViewToolBars'
     Caption = 'ToolBar &1'
     Checked = True
     ImageIndex = 1
     OnExecute =
ViewToolBar1Execute
   end
   <-------> Object  ViewToolBar2:
TAction
     Category = 'ViewToolBars'
     Caption = 'ToolBar &2'
     Checked = True
     ImageIndex = 2
     OnExecute =
ViewToolBar2Execute
   end
   <-------> Object
ViewWhiteWindow: TAction
     Category = 'ViewWindows'
     Caption = '&White'
     Hint = 'View white window'
     OnExecute =
ViewWhiteWindowExecute
   end
   <-------> Object  ExitAction:
TAction
     Caption = 'E&xit'
     OnExecute = ExitActionExecute
   end
   <-------> Object
ViewBlueWindow: TAction
     Tag = 1
     Category = 'ViewWindows'
     Caption = '&Blue'
     OnExecute =
ViewWhiteWindowExecute
```

```
TToolButton
        Left = 172
        Top = 0
        Action = ViewPurpleWindow
      end
      <-------> Object  ToolButton7:
TToolButton
        Left = 215
        Top = 0
        Action = ViewRedWindow
      end
      <-------> Object  ToolButton4:
TToolButton
        Left = 258
        Top = 0
        Action = ViewTealWindow
      end
    end
  end
  <-------> Object  LeftDockPanel:
TPanel
    Left = 4
    Top = 52
    Width = 0
    Height = 388
    Align = alLeft
    DockSite = True
    TabOrder = 1
    OnDockDrop =
LeftDockPanelDockDrop
    OnDockOver =
LeftDockPanelDockOver
    OnGetSiteInfo =
LeftDockPanelGetSiteInfo
    OnUnDock =
LeftDockPanelUnDock
  end
  <-------> Object  BottomDockPanel:
TPanel
    Left = 0
    Top = 440
    Width = 675
    Height = 0
    Align = alBottom
    DockSite = True
    TabOrder = 2
    OnDockDrop =
LeftDockPanelDockDrop
```

<-------> Object Exit2: TMenuItem
Action = ExitAction
end
end
<-------> Object View2: TMenuItem
Caption = '&View'
<-------> Object ToolBar11: TMenuItem
Action = ViewToolBar1
end
<-------> Object ToolBar21: TMenuItem
Action = ViewToolBar2
end
<-------> Object N2: TMenuItem
Caption = '-'
end
<-------> Object White1: TMenuItem
Action = ViewWhiteWindow
end
<-------> Object Blue1: TMenuItem
Action = ViewBlueWindow
end
<-------> Object Green1: TMenuItem
Action = ViewGreenWindow
end
<-------> Object Lime1: TMenuItem
Action = ViewLimeWindow
end
<-------> Object Purple1: TMenuItem
Action = ViewPurpleWindow
end
<-------> Object Red1: TMenuItem
Action = ViewRedWindow
end
<-------> Object Teal1: TMenuItem
Action = ViewTealWindow
end
end
end

end
<-------> Object ViewGreenWindow: TAction
Tag = 2
Category = 'ViewWindows'
Caption = '&Green'
OnExecute = ViewWhiteWindowExecute
end
<-------> Object ViewRedWindow: TAction
Tag = 3
Category = 'ViewWindows'
Caption = '&Red'
OnExecute = ViewWhiteWindowExecute
end
<-------> Object ViewTealWindow: TAction
Tag = 4
Category = 'ViewWindows'
Caption = '&Teal'
OnExecute = ViewWhiteWindowExecute
end
<-------> Object ViewPurpleWindow: TAction
Tag = 5
Category = 'ViewWindows'
Caption = '&Purple'
OnExecute = ViewWhiteWindowExecute
end
<-------> Object ViewLimeWindow: TAction
Tag = 6
Category = 'ViewWindows'
Caption = '&Lime'
OnExecute = ViewWhiteWindowExecute
end
end
<-------> Object MainMenu1: TMainMenu
Left = 176
Top = 80
<-------> Object File2: TMenuItem
Caption = '&File'

Procedures90
الملف التنفيذي لبرنامج
(RichEdit)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
unit remain;

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, Buttons, ExtCtrls, Menus, ComCtrls, ClipBrd,
  ToolWin, ActnList, ImgList;

type
  TMainForm = class(TForm)
    MainMenu: TMainMenu;
    FileNewItem: TMenuItem;
    FileOpenItem: TMenuItem;
    FileSaveItem: TMenuItem;
    FileSaveAsItem: TMenuItem;
    FilePrintItem: TMenuItem;
    FileExitItem: TMenuItem;
    EditUndoItem: TMenuItem;
    EditCutItem: TMenuItem;
    EditCopyItem: TMenuItem;
    EditPasteItem: TMenuItem;
    HelpAboutItem: TMenuItem;
    OpenDialog: TOpenDialog;
    SaveDialog: TSaveDialog;
    PrintDialog: TPrintDialog;
    Ruler: TPanel;
    FontDialog1: TFontDialog;
    FirstInd: TLabel;
    LeftInd: TLabel;
    RulerLine: TBevel;
    RightInd: TLabel;
    N5: TMenuItem;
    miEditFont: TMenuItem;
    Editor: TRichEdit;
    StatusBar: TStatusBar;
    StandardToolBar: TToolBar;
    OpenButton: TToolButton;
    SaveButton: TToolButton;
    PrintButton: TToolButton;
    ToolButton5: TToolButton;
    UndoButton: TToolButton;
    CutButton: TToolButton;
    CopyButton: TToolButton;
    PasteButton: TToolButton;
    ToolButton10: TToolButton;
    FontName: TComboBox;
    FontSize: TEdit;
    ToolButton11: TToolButton;
    UpDown1: TUpDown;
    BoldButton: TToolButton;
    ItalicButton: TToolButton;
    UnderlineButton: TToolButton;
    ToolButton16: TToolButton;
    LeftAlign: TToolButton;
    CenterAlign: TToolButton;
    RightAlign: TToolButton;
    ToolButton20: TToolButton;
    BulletsButton: TToolButton;
    ToolbarImages: TImageList;
    ActionList1: TActionList;
    FileNewCmd: TAction;
    FileOpenCmd: TAction;
    FileSaveCmd: TAction;
    FilePrintCmd: TAction;
    FileExitCmd: TAction;
    ToolButton1: TToolButton;
    ToolButton2: TToolButton;
    Bevel1: TBevel;
    LanguageMenu: TMenuItem;
    LanguageEnglish: TMenuItem;
    LanguageGerman: TMenuItem;
    EditCutCmd: TAction;
    EditCopyCmd: TAction;
    EditPasteCmd: TAction;
    EditUndoCmd: TAction;
    EditFontCmd: TAction;
    FileSaveAsCmd: TAction;
    LanguageFrench: TMenuItem;

    <-------> procedure
SelectionChange(Sender: TObject);
```

BulletsButtonClick(Sender: TObject);
<-------> procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
<-------> procedure RulerItemMouseDown(Sender: TObject; Button: TMouseButton;
Shift: TShiftState; X, Y: Integer);
<-------> procedure RulerItemMouseMove(Sender: TObject; Shift: TShiftState; X,
Y: Integer);
<-------> procedure FirstIndMouseUp(Sender: TObject; Button: TMouseButton;
Shift: TShiftState; X, Y: Integer);
<-------> procedure LeftIndMouseUp(Sender: TObject; Button: TMouseButton;
Shift: TShiftState; X, Y: Integer);
<-------> procedure RightIndMouseUp(Sender: TObject; Button: TMouseButton;
Shift: TShiftState; X, Y: Integer);
<-------> procedure FormShow(Sender: TObject);
<-------> procedure RichEditChange(Sender: TObject);
<-------> procedure SwitchLanguage(Sender: TObject);
<-------> procedure ActionList2Update(Action: TBasicAction;
var Handled: Boolean);
private
FFileName: string;
FUpdating: Boolean;
FDragOfs: Integer;
FDragging: Boolean;
function CurrText: TTextAttributes;
<-------> procedure GetFontNames;
<-------> procedure

<-------> procedure FormCreate(Sender: TObject);
<-------> procedure ShowHint(Sender: TObject);
<-------> procedure FileNew(Sender: TObject);
<-------> procedure FileOpen(Sender: TObject);
<-------> procedure FileSave(Sender: TObject);
<-------> procedure FileSaveAs(Sender: TObject);
<-------> procedure FilePrint(Sender: TObject);
<-------> procedure FileExit(Sender: TObject);
<-------> procedure EditUndo(Sender: TObject);
<-------> procedure EditCut(Sender: TObject);
<-------> procedure EditCopy(Sender: TObject);
<-------> procedure EditPaste(Sender: TObject);
<-------> procedure HelpAbout(Sender: TObject);
<-------> procedure SelectFont(Sender: TObject);
<-------> procedure RulerResize(Sender: TObject);
<-------> procedure FormResize(Sender: TObject);
<-------> procedure FormPaint(Sender: TObject);
<-------> procedure BoldButtonClick(Sender: TObject);
<-------> procedure ItalicButtonClick(Sender: TObject);
<-------> procedure FontSizeChange(Sender: TObject);
<-------> procedure AlignButtonClick(Sender: TObject);
<-------> procedure FontNameChange(Sender: TObject);
<-------> procedure UnderlineButtonClick(Sender: TObject);
<-------> procedure

```
LANG_GERMAN;

{$R *.dfm}

<-------> procedure TMainForm.SelectionChange(Sender: TObject);
begin
  with Editor.Paragraph do
  try
    FUpdating := True;
    FirstInd.Left := Trunc(FirstIndent*RulerAdj)-4+GutterWid;
    LeftInd.Left := Trunc((LeftIndent+FirstIndent)*RulerAdj)-4+GutterWid;
    RightInd.Left := Ruler.ClientWidth-6-Trunc((RightIndent+GutterWid)*RulerAdj);
    BoldButton.Down := fsBold in Editor.SelAttributes.Style;
    ItalicButton.Down := fsItalic in Editor.SelAttributes.Style;
    UnderlineButton.Down := fsUnderline in Editor.SelAttributes.Style;
    BulletsButton.Down := Boolean(Numbering);
    FontSize.Text := IntToStr(Editor.SelAttributes.Size);
    FontName.Text := Editor.SelAttributes.Name;
    case Ord(Alignment) of
      0: LeftAlign.Down := True;
      1: RightAlign.Down := True;
      2: CenterAlign.Down := True;
    end;
    UpdateCursorPos;
  finally
    FUpdating := False;
  end;
end;

function TMainForm.CurrText: TTextAttributes;
begin
  if Editor.SelLength > 0 then Result
```

```
SetFileName(const FileName: String);
    <-------> procedure CheckFileSave;
    <-------> procedure SetupRuler;
    <-------> procedure SetEditRect;
    <-------> procedure UpdateCursorPos;
    <-------> procedure WMDropFiles(var Msg: TWMDropFiles); message WM_DROPFILES;
    <-------> procedure PerformFileOpen(const AFileName: string);
    <-------> procedure SetModified(Value: Boolean);
  end;

var
  MainForm: TMainForm;

implementation

uses REAbout, RichEdit, ShellAPI, ReInit;

resourcestring
  sSaveChanges = 'Save changes to %s?';
  sOverWrite = 'OK to overwrite %s';
  sUntitled = 'Untitled';
  sModified = 'Modified';
  sColRowInfo = 'Line: %3d   Col: %3d';

const
  RulerAdj = 4/3;
  GutterWid = 6;

  ENGLISH = (SUBLANG_ENGLISH_US shl 10) or LANG_ENGLISH;
  FRENCH  = (SUBLANG_FRENCH shl 10) or LANG_FRENCH;
  GERMAN  = (SUBLANG_GERMAN shl 10) or
```

```
mbYesNoCancel, 0);
  case SaveResp of
    idYes: FileSave(Self);
    idNo: {Nothing};
    idCancel: Abort;
  end;
end;

<-------> procedure
TMainForm.SetupRuler;
var
  I: Integer;
  S: String;
begin
  SetLength(S, 201);
  I := 1;
  while I < 200 do
  begin
    S[I] := #9;
    S[I+1] := '|';
    Inc(I, 2);
  end;
  Ruler.Caption := S;
end;

<-------> procedure
TMainForm.SetEditRect;
var
  R: TRect;
begin
  with Editor do
  begin
    R := Rect(GutterWid, 0,
ClientWidth-GutterWid,
ClientHeight);
    SendMessage(Handle,
EM_SETRECT, 0, Longint(@R));
  end;
end;

{ Event Handlers }

<-------> procedure
TMainForm.FormCreate(Sender:
TObject);
begin
  Application.OnHint := ShowHint;
  OpenDialog.InitialDir :=
```

```
:= Editor.SelAttributes
  else Result := Editor.DefAttributes;
end;

function EnumFontsProc(var
LogFont: TLogFont; var
TextMetric: TTextMetric;
  FontType: Integer; Data: Pointer):
Integer; stdcall;
begin

TStrings(Data).Add(LogFont.lfFace
Name);
  Result := 1;
end;

<-------> procedure
TMainForm.GetFontNames;
var
  DC: HDC;
begin
  DC := GetDC(0);
  EnumFonts(DC, nil,
@EnumFontsProc,
Pointer(FontName.Items));
  ReleaseDC(0, DC);
  FontName.Sorted := True;
end;

<-------> procedure
TMainForm.SetFileName(const
FileName: String);
begin
  FFileName := FileName;
  Caption := Format('%s - %s',
[ExtractFileName(FileName),
Application.Title]);
end;

<-------> procedure
TMainForm.CheckFileSave;
var
  SaveResp: Integer;
begin
  if not Editor.Modified then Exit;
  SaveResp :=
MessageDlg(Format(sSaveChanges,
[FFileName]),
    mtConfirmation,
```

```
  SetFileName(sUntitled);
  Editor.Lines.Clear;
  Editor.Modified := False;
  SetModified(False);
end;

<-------> procedure TMainForm.PerformFileOpen(const AFileName: string);
begin
  Editor.Lines.LoadFromFile(AFileName);
  SetFileName(AFileName);
  Editor.SetFocus;
  Editor.Modified := False;
  SetModified(False);
end;

<-------> procedure TMainForm.FileOpen(Sender: TObject);
begin
  CheckFileSave;
  if OpenDialog.Execute then
  begin
    PerformFileOpen(OpenDialog.FileName);
    Editor.ReadOnly := ofReadOnly in OpenDialog.Options;
  end;
end;

<-------> procedure TMainForm.FileSave(Sender: TObject);
begin
  if FFileName = sUntitled then
    FileSaveAs(Sender)
  else
  begin
    Editor.Lines.SaveToFile(FFileName);
    Editor.Modified := False;
    SetModified(False);
  end;
end;
```

```
ExtractFilePath(ParamStr(0));
  SaveDialog.InitialDir := OpenDialog.InitialDir;
  SetFileName(sUntitled);
  GetFontNames;
  SetupRuler;
  SelectionChange(Self);

  CurrText.Name := DefFontData.Name;
  CurrText.Size := -MulDiv(DefFontData.Height, 72, Screen.PixelsPerInch);

  LanguageEnglish.Tag := ENGLISH;
  LanguageFrench.Tag := FRENCH;
  LanguageGerman.Tag := GERMAN;
  case SysLocale.DefaultLCID of
    ENGLISH: SwitchLanguage(LanguageEnglish);
    FRENCH: SwitchLanguage(LanguageFrench);
    GERMAN: SwitchLanguage(LanguageGerman);
  end;
end;

<-------> procedure TMainForm.ShowHint(Sender: TObject);
begin
  if Length(Application.Hint) > 0 then
  begin
    StatusBar.SimplePanel := True;
    StatusBar.SimpleText := Application.Hint;
  end
  else StatusBar.SimplePanel := False;
end;

<-------> procedure TMainForm.FileNew(Sender: TObject);
begin
```

```
end;

<-------> procedure
TMainForm.EditCut(Sender:
TObject);
begin
  Editor.CutToClipboard;
end;

<-------> procedure
TMainForm.EditCopy(Sender:
TObject);
begin
  Editor.CopyToClipboard;
end;

<-------> procedure
TMainForm.EditPaste(Sender:
TObject);
begin
  Editor.PasteFromClipboard;
end;

<-------> procedure
TMainForm.HelpAbout(Sender:
TObject);
begin
  with TAboutBox.Create(Self) do
  try
    ShowModal;
  finally
    Free;
  end;
end;

<-------> procedure
TMainForm.SelectFont(Sender:
TObject);
begin

FontDialog1.Font.Assign(Editor.Sel
Attributes);
  if FontDialog1.Execute then

CurrText.Assign(FontDialog1.Font);
  SelectionChange(Self);
  Editor.SetFocus;
end;


<-------> procedure
TMainForm.FileSaveAs(Sender:
TObject);
begin
  if SaveDialog.Execute then
  begin
    if
FileExists(SaveDialog.FileName)
then
      if
MessageDlg(Format(sOverWrite,
[SaveDialog.FileName]),
        mtConfirmation,
mbYesNoCancel, 0) <> idYes then
Exit;

Editor.Lines.SaveToFile(SaveDialog.
FileName);

SetFileName(SaveDialog.FileName);
    Editor.Modified := False;
    SetModified(False);
  end;
end;

<-------> procedure
TMainForm.FilePrint(Sender:
TObject);
begin
  if PrintDialog.Execute then
    Editor.Print(FFileName);
end;

<-------> procedure
TMainForm.FileExit(Sender:
TObject);
begin
  Close;
end;

<-------> procedure
TMainForm.EditUndo(Sender:
TObject);
begin
  with Editor do
    if HandleAllocated then
SendMessage(Handle, EM_UNDO,
0, 0);
```

```
<-------> procedure TMainForm.FontSizeChange(Sender: TObject);
begin
  if FUpdating then Exit;
  CurrText.Size := StrToInt(FontSize.Text);
end;

<-------> procedure TMainForm.AlignButtonClick(Sender: TObject);
begin
  if FUpdating then Exit;
  Editor.Paragraph.Alignment := TAlignment(TControl(Sender).Tag);
end;

<-------> procedure TMainForm.FontNameChange(Sender: TObject);
begin
  if FUpdating then Exit;
  CurrText.Name := FontName.Items[FontName.ItemIndex];
end;

<-------> procedure TMainForm.UnderlineButtonClick(Sender: TObject);
begin
  if FUpdating then Exit;
  if UnderlineButton.Down then
    CurrText.Style := CurrText.Style + [fsUnderline]
  else
    CurrText.Style := CurrText.Style - [fsUnderline];
end;

<-------> procedure TMainForm.BulletsButtonClick(Sender: TObject);
begin
  if FUpdating then Exit;
  Editor.Paragraph.Numbering := TNumberingStyle(BulletsButton.Do

<-------> procedure TMainForm.RulerResize(Sender: TObject);
begin
  RulerLine.Width := Ruler.ClientWidth - (RulerLine.Left*2);
end;

<-------> procedure TMainForm.FormResize(Sender: TObject);
begin
  SetEditRect;
  SelectionChange(Sender);
end;

<-------> procedure TMainForm.FormPaint(Sender: TObject);
begin
  SetEditRect;
end;

<-------> procedure TMainForm.BoldButtonClick(Sender: TObject);
begin
  if FUpdating then Exit;
  if BoldButton.Down then
    CurrText.Style := CurrText.Style + [fsBold]
  else
    CurrText.Style := CurrText.Style - [fsBold];
end;

<-------> procedure TMainForm.ItalicButtonClick(Sender: TObject);
begin
  if FUpdating then Exit;
  if ItalicButton.Down then
    CurrText.Style := CurrText.Style + [fsItalic]
  else
    CurrText.Style := CurrText.Style - [fsItalic];
end;
```

```
GutterWid) / RulerAdj);
  LeftIndMouseUp(Sender, Button, Shift, X, Y);
end;

<-------> procedure TMainForm.LeftIndMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  FDragging := False;
  Editor.Paragraph.LeftIndent := Trunc((LeftInd.Left+FDragOfs-GutterWid) / RulerAdj)-Editor.Paragraph.FirstIndent;
  SelectionChange(Sender);
end;

<-------> procedure TMainForm.RightIndMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  FDragging := False;
  Editor.Paragraph.RightIndent := Trunc((Ruler.ClientWidth-RightInd.Left+FDragOfs-2) / RulerAdj)-2*GutterWid;
  SelectionChange(Sender);
end;

<-------> procedure TMainForm.UpdateCursorPos;
var
  CharPos: TPoint;
begin
  CharPos.Y := SendMessage(Editor.Handle, EM_EXLINEFROMCHAR, 0,
    Editor.SelStart);
  CharPos.X := (Editor.SelStart -
    SendMessage(Editor.Handle, EM_LINEINDEX, CharPos.Y, 0));
  Inc(CharPos.Y);
  Inc(CharPos.X);
  StatusBar.Panels[0].Text := Format(sColRowInfo, [CharPos.Y, 
wn]);
end;

<-------> procedure TMainForm.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
begin
  try
    CheckFileSave;
  except
    CanClose := False;
  end;
end;

{ Ruler Indent Dragging }

<-------> procedure TMainForm.RulerItemMouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  FDragOfs := (TLabel(Sender).Width div 2);
  TLabel(Sender).Left := TLabel(Sender).Left+X-FDragOfs;
  FDragging := True;
end;

<-------> procedure TMainForm.RulerItemMouseMove(Sender: TObject; Shift: TShiftState;
  X, Y: Integer);
begin
  if FDragging then
    TLabel(Sender).Left := TLabel(Sender).Left+X-FDragOfs
end;

<-------> procedure TMainForm.FirstIndMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  FDragging := False;
  Editor.Paragraph.FirstIndent := Trunc((FirstInd.Left+FDragOfs-
```

```
  if Value then
StatusBar.Panels[1].Text :=
sModified
  else StatusBar.Panels[1].Text := '';
end;

<-------> procedure
TMainForm.SwitchLanguage(Sende
r: TObject);
var
  Name : String;
  Size : Integer;
begin
  if
LoadNewResourceModule(TCompo
nent(Sender).Tag) <> 0 then
  begin
    Name := FontName.Text;
    Size := StrToInt(FontSize.Text);
    ReinitializeForms;
    LanguageEnglish.Checked :=
LanguageEnglish = Sender;
    LanguageFrench.Checked  :=
LanguageFrench  = Sender;
    LanguageGerman.Checked  :=
LanguageGerman  = Sender;

    CurrText.Name := Name;
    CurrText.Size := Size;
    SelectionChange(Self);
    FontName.SelLength := 0;

    SetupRuler;
    if Visible then Editor.SetFocus;
  end;
end;

<-------> procedure
TMainForm.ActionList2Update(Acti
on: TBasicAction;
  var Handled: Boolean);
begin
 { Update the status of the edit
commands }
  EditCutCmd.Enabled :=
Editor.SelLength > 0;
  EditCopyCmd.Enabled :=
EditCutCmd.Enabled;
  if Editor.HandleAllocated then
```

```
CharPos.X]);
end;

<-------> procedure
TMainForm.FormShow(Sender:
TObject);
begin
  UpdateCursorPos;
  DragAcceptFiles(Handle, True);
  RichEditChange(nil);
  Editor.SetFocus;
  { Check if we should load a file
from the command line }
  if (ParamCount > 0) and
FileExists(ParamStr(1)) then
    PerformFileOpen(ParamStr(1));
end;

<-------> procedure
TMainForm.WMDropFiles(var
Msg: TWMDropFiles);
var
  CFileName: array[0..MAX_PATH]
of Char;
begin
  try
    if DragQueryFile(Msg.Drop, 0,
CFileName, MAX_PATH) > 0 then
    begin
      CheckFileSave;
      PerformFileOpen(CFileName);
      Msg.Result := 0;
    end;
  finally
    DragFinish(Msg.Drop);
  end;
end;

<-------> procedure
TMainForm.RichEditChange(Sende
r: TObject);
begin
  SetModified(Editor.Modified);
end;

<-------> procedure
TMainForm.SetModified(Value:
Boolean);
begin
```

```
<-------> procedure
TAboutBox.FormCreate(Sender:
TObject);
var
  MS: TMemoryStatus;
begin
  GlobalMemoryStatus(MS);
  PhysMem.Caption :=
FormatFloat('#,###" KB"',
MS.dwTotalPhys / 1024);
  FreeRes.Caption := Format('%d
%%', [MS.dwMemoryLoad]);
end;

end.

\\\\\\\\\\
```

<-------> Objects87
الملف النصي لبرنامج
(RichEdit)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
<-------> Object MainForm:
TMainForm
  Left = 173
  Top = 156
  Width = 625
  Height = 209
  ActiveControl = Editor
  Caption = 'Rich Edit Control
Demo'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'Default'
  Font.Style = []
  Icon.Data = {
FF}
  Menu = MainMenu
  OldCreateOrder = True
```

```
  begin
    EditUndoCmd.Enabled :=
Editor.Perform(EM_CANUNDO, 0,
0) <> 0;
    EditPasteCmd.Enabled :=
Editor.Perform(EM_CANPASTE, 0,
0) <> 0;
  end;
end;

end.

\\\\\\\\\\\\\\\\\\\\

unit reabout;

interface

uses Windows, Classes, Graphics,
Forms, Controls, StdCtrls,
  Buttons, ExtCtrls, SysUtils;

type
  TAboutBox = class(TForm)
    OKButton: TButton;
    ProgramIcon: TImage;
    Label1: TLabel;
    Bevel1: TBevel;
    Label2: TLabel;
    Label3: TLabel;
    PhysMem: TLabel;
    Label4: TLabel;
    FreeRes: TLabel;
    <-------> procedure
FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  AboutBox: TAboutBox;

implementation

{$R *.dfm}

```

Left = 2
Top = 12
Width = 10
Height = 11
AutoSize = False
Caption = 'é'
DragCursor = crArrow
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'Wingdings'
Font.Style = []
ParentFont = False
OnMouseDown = RulerItemMouseDown
OnMouseMove = RulerItemMouseMove
OnMouseUp = LeftIndMouseUp
end
<-------> Object RulerLine: TBevel
Left = 4
Top = 12
Width = 579
Height = 2
Shape = bsTopLine
end
<-------> Object RightInd: TLabel
Left = 575
Top = 14
Width = 9
Height = 12
Caption = 'ñ'
DragCursor = crArrow
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'Wingdings'
Font.Style = []
ParentFont = False
OnMouseDown = RulerItemMouseDown
OnMouseMove = RulerItemMouseMove
OnMouseUp = RightIndMouseUp

Position = poDefaultSizeOnly
OnCloseQuery = FormCloseQuery
OnCreate = FormCreate
OnPaint = FormPaint
OnResize = FormResize
OnShow = FormShow
PixelsPerInch = 96
TextHeight = 13
<-------> Object Ruler: TPanel
Left = 0
Top = 32
Width = 617
Height = 26
Align = alTop
Alignment = taLeftJustify
BevelOuter = bvNone
Caption = 'asdf'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'Arial'
Font.Style = []
ParentFont = False
TabOrder = 0
OnResize = RulerResize
<-------> Object FirstInd: TLabel
Left = 2
Top = 2
Width = 10
Height = 9
AutoSize = False
Caption = 'ê'
DragCursor = crArrow
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'Wingdings'
Font.Style = []
ParentFont = False
OnMouseDown = RulerItemMouseDown
OnMouseMove = RulerItemMouseMove
OnMouseUp = FirstIndMouseUp
end
<-------> Object LeftInd: TLabel

```
   Images = ToolbarImages
   Indent = 4
   ParentShowHint = False
   ShowHint = True
   TabOrder = 3
   Wrapable = False
   <-------> Object ToolButton1: TToolButton
     Left = 4
     Top = 0
     Action = FileNewCmd
   end
   <-------> Object OpenButton: TToolButton
     Left = 27
     Top = 0
     Action = FileOpenCmd
   end
   <-------> Object SaveButton: TToolButton
     Left = 50
     Top = 0
     Action = FileSaveCmd
   end
   <-------> Object PrintButton: TToolButton
     Left = 73
     Top = 0
     Action = FilePrintCmd
   end
   <-------> Object ToolButton5: TToolButton
     Left = 96
     Top = 0
     Width = 8
     ImageIndex = 3
     Style = tbsDivider
   end
   <-------> Object CutButton: TToolButton
     Left = 104
     Top = 0
     Action = EditCutCmd
   end
   <-------> Object CopyButton: TToolButton
     Left = 127
     Top = 0
   end
   <-------> Object Bevel1: TBevel
     Left = 0
     Top = 0
     Width = 617
     Height = 2
     Align = alTop
     Shape = bsTopLine
   end
  end
  <-------> Object Editor: TRichEdit
   Left = 0
   Top = 58
   Width = 617
   Height = 78
   Align = alClient
   ScrollBars = ssBoth
   TabOrder = 1
   OnChange = RichEditChange
   OnSelectionChange = SelectionChange
  end
  <-------> Object StatusBar: TStatusBar
   Left = 0
   Top = 136
   Width = 617
   Height = 19
   Panels = <
     item
       Width = 120
     end
     item
       Alignment = taCenter
       Width = 60
     end
     item
       Width = 50
     end>
  end
  <-------> Object StandardToolBar: TToolBar
   Left = 0
   Top = 0
   Width = 617
   AutoSize = True
   BorderWidth = 2
   Flat = True
```

Width = 26
Height = 22
Hint = 'Font Size|Select font size'
TabOrder = 1
Text = '0'
OnChange = FontSizeChange
end
<-------> Object UpDown1: TUpDown
Left = 411
Top = 0
Width = 15
Height = 22
Associate = FontSize
TabOrder = 2
end
<-------> Object ToolButton2: TToolButton
Left = 426
Top = 0
Width = 8
Caption = 'ToolButton2'
ImageIndex = 15
Style = tbsSeparator
end
<-------> Object BoldButton: TToolButton
Left = 434
Top = 0
Hint = 'Bold'
ImageIndex = 8
Style = tbsCheck
OnClick = BoldButtonClick
end
<-------> Object ItalicButton: TToolButton
Left = 457
Top = 0
Hint = 'Italic'
ImageIndex = 9
Style = tbsCheck
OnClick = ItalicButtonClick
end
<-------> Object UnderlineButton: TToolButton
Left = 480
Top = 0
Hint = 'Underline'

Action = EditCopyCmd
end
<-------> Object PasteButton: TToolButton
Left = 150
Top = 0
Action = EditPasteCmd
end
<-------> Object UndoButton: TToolButton
Left = 173
Top = 0
Action = EditUndoCmd
end
<-------> Object ToolButton10: TToolButton
Left = 196
Top = 0
Width = 8
ImageIndex = 7
Style = tbsSeparator
end
<-------> Object FontName: TComboBox
Left = 204
Top = 0
Width = 173
Height = 21
Hint = 'Font Name|Select font name'
Ctl3D = False
DropDownCount = 10
ItemHeight = 13
ParentCtl3D = False
TabOrder = 0
OnChange = FontNameChange
end
<-------> Object ToolButton11: TToolButton
Left = 377
Top = 0
Width = 8
ImageIndex = 8
Style = tbsSeparator
end
<-------> Object FontSize: TEdit
Left = 385
Top = 0

```
    Width = 8
    ImageIndex = 15
    Style = tbsDivider
  end
  <-------> Object BulletsButton: TToolButton
    Left = 588
    Top = 0
    Hint = 'Bullets|Enter bullet mode'
    ImageIndex = 14
    Style = tbsCheck
    OnClick = BulletsButtonClick
  end
 end
 <-------> Object MainMenu: TMainMenu
  Images = ToolbarImages
  Left = 120
  Top = 68
  <-------> Object FileMenu: TMenuItem
    Caption = '&File'
    <-------> Object FileNewItem: TMenuItem
      Action = FileNewCmd
    end
    <-------> Object FileOpenItem: TMenuItem
      Action = FileOpenCmd
    end
    <-------> Object FileSaveItem: TMenuItem
      Action = FileSaveCmd
    end
    <-------> Object FileSaveAsItem: TMenuItem
      Action = FileSaveAsCmd
    end
    <-------> Object N1: TMenuItem
      Caption = '-'
    end
    <-------> Object FilePrintItem: TMenuItem
      Action = FilePrintCmd
    end
    <-------> Object N4: TMenuItem
      Caption = '-'
```

```
    ImageIndex = 10
    Style = tbsCheck
    OnClick = UnderlineButtonClick
  end
  <-------> Object ToolButton16: TToolButton
    Left = 503
    Top = 0
    Width = 8
    ImageIndex = 12
    Style = tbsDivider
  end
  <-------> Object LeftAlign: TToolButton
    Left = 511
    Top = 0
    Hint = 'Align Left'
    Grouped = True
    ImageIndex = 11
    Style = tbsCheck
    OnClick = AlignButtonClick
  end
  <-------> Object CenterAlign: TToolButton
    Tag = 2
    Left = 534
    Top = 0
    Hint = 'Center'
    Grouped = True
    ImageIndex = 12
    Style = tbsCheck
    OnClick = AlignButtonClick
  end
  <-------> Object RightAlign: TToolButton
    Tag = 1
    Left = 557
    Top = 0
    Hint = 'Align Right'
    Grouped = True
    ImageIndex = 13
    Style = tbsCheck
    OnClick = AlignButtonClick
  end
  <-------> Object ToolButton20: TToolButton
    Left = 580
    Top = 0
```

```
      Caption = '&French'
      Hint = 'Switch translation to French'
      OnClick = SwitchLanguage
    end
    <-------> Object LanguageGerman: TMenuItem
      Caption = '&German'
      Hint = 'Switch translation to German'
      OnClick = SwitchLanguage
    end
  end
  <-------> Object HelpMenu: TMenuItem
    Caption = '&Help'
    <-------> Object HelpAboutItem: TMenuItem
      Caption = '&About...'
      Hint = 'Show program information'
      OnClick = HelpAbout
    end
  end
 end
 <-------> Object OpenDialog: TOpenDialog
  Filter = 'Rich Text Files (*.RTF)|*.RTF|Text Files (*.TXT)|*.TXT'
  Left = 56
  Top = 104
 end
 <-------> Object SaveDialog: TSaveDialog
  Filter = 'Rich Text Files (*.RTF)|*.RTF|Text Files (*.TXT)|*.TXT'
  Left = 88
  Top = 104
 end
 <-------> Object PrintDialog: TPrintDialog
  Left = 120
  Top = 104
 end
 <-------> Object FontDialog1: TFontDialog
  Font.Charset =
```

```
    end
    <-------> Object FileExitItem: TMenuItem
      Action = FileExitCmd
    end
  end
  <-------> Object EditMenu: TMenuItem
    Caption = '&Edit'
    <-------> Object EditUndoItem: TMenuItem
      Action = EditUndoCmd
    end
    <-------> Object N2: TMenuItem
      Caption = '-'
    end
    <-------> Object EditCutItem: TMenuItem
      Action = EditCutCmd
    end
    <-------> Object EditCopyItem: TMenuItem
      Action = EditCopyCmd
    end
    <-------> Object EditPasteItem: TMenuItem
      Action = EditPasteCmd
    end
    <-------> Object N5: TMenuItem
      Caption = '-'
    end
    <-------> Object miEditFont: TMenuItem
      Caption = '&Font...'
      OnClick = SelectFont
    end
  end
  <-------> Object LanguageMenu: TMenuItem
    Caption = '&Language'
    <-------> Object LanguageEnglish: TMenuItem
      Caption = '&English'
      Hint = 'Switch translation to English'
      OnClick = SwitchLanguage
    end
    <-------> Object LanguageFrench: TMenuItem
```

TAction
Category = 'File'
Caption = '&Print'
Hint = 'Print current file'
ImageIndex = 3
ShortCut = 16464
OnExecute = FilePrint
end
<-------> Object FileExitCmd: TAction
Category = 'File'
Caption = 'E&xit'
Hint = 'Exit this application'
ShortCut = 32856
OnExecute = FileExit
end
<-------> Object FileSaveAsCmd: TAction
Category = 'File'
Caption = 'Save &As...'
Hint = 'Save current file under a new name'
OnExecute = FileSaveAs
end
end
<-------> Object ActionList2: TActionList
Images = ToolbarImages
OnUpdate = ActionList2Update
Left = 56
Top = 68
<-------> Object EditUndoCmd: TAction
Category = 'Edit'
Caption = '&Undo'
Hint = 'Undo the last action'
ImageIndex = 4
ShortCut = 16474
OnExecute = EditUndo
end
<-------> Object EditCutCmd: TAction
Category = 'Edit'
Caption = 'Cu&t'
Hint = 'Delete selected item'
ImageIndex = 5
ShortCut = 16472
OnExecute = EditCut

DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
Left = 24
Top = 104
end
<-------> Object ToolbarImages: TImageList
Left = 88
Top = 68
Bitmap = {
000000000000}
end
<-------> Object ActionList1: TActionList
Images = ToolbarImages
Left = 24
Top = 68
<-------> Object FileNewCmd: TAction
Category = 'File'
Caption = '&New'
Hint = 'Create a new file'
ImageIndex = 0
ShortCut = 16462
OnExecute = FileNew
end
<-------> Object FileOpenCmd: TAction
Category = 'File'
Caption = '&Open...'
Hint = 'Open an existing file'
ImageIndex = 1
ShortCut = 16463
OnExecute = FileOpen
end
<-------> Object FileSaveCmd: TAction
Category = 'File'
Caption = '&Save'
Hint = 'Save current file'
ImageIndex = 2
ShortCut = 16467
OnExecute = FileSave
end
<-------> Object FilePrintCmd:

Position = poScreenCenter
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> Object ProgramIcon: TImage
Left = 8
Top = 12
Width = 64
Height = 64
AutoSize = True
Picture.Data = {
BFBF}
Transparent = True
IsControl = True
end
<-------> Object Label1: TLabel
Left = 81
Top = 22
Width = 143
Height = 13
Caption = 'Delphi Rich Edit Control Demo'
end
<-------> Object Bevel1: TBevel
Left = 81
Top = 134
Width = 260
Height = 2
Shape = bsTopLine
end
<-------> Object Label2: TLabel
Left = 81
Top = 45
Width = 183
Height = 13
Caption = 'Copyright © 1998 Borland International'
end
<-------> Object Label3: TLabel
Left = 81
Top = 145
Width = 187
Height = 13
Caption = 'Physical Memory Available to Windows:'
end
<-------> Object PhysMem: TLabel

end
<-------> Object EditCopyCmd: TAction
Category = 'Edit'
Caption = '&Copy'
Hint = 'Copy selected item to clipboard'
ImageIndex = 6
ShortCut = 16451
OnExecute = EditCopy
end
<-------> Object EditPasteCmd: TAction
Category = 'Edit'
Caption = '&Paste'
Hint = 'Paste contents of clipboard'
ImageIndex = 7
ShortCut = 16470
OnExecute = EditPaste
end
<-------> Object EditFontCmd: TAction
Category = 'Edit'
Caption = '&Font...'
OnExecute = SelectFont
end
end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
<-------> Object AboutBox: TAboutBox
Left = 183
Top = 438
BorderStyle = bsDialog
Caption = 'About RichEdit'
ClientHeight = 243
ClientWidth = 357
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True

uses
SysUtils, Windows, Messages, Classes, Graphics, Controls, Forms, Dialogs,
Menus, StdCtrls, ComCtrls;
type
TEditForm = class(TForm)
MainMenu1: TMainMenu;
File1: TMenuItem;
New1: TMenuItem;
Open1: TMenuItem;
Save1: TMenuItem;
Saveas1: TMenuItem;
Print1: TMenuItem;
N2: TMenuItem;
Exit1: TMenuItem;
Edit1: TMenuItem;
Cut1: TMenuItem;
Copy1: TMenuItem;
Paste1: TMenuItem;
Delete1: TMenuItem;
N3: TMenuItem;
Selectall1: TMenuItem;
Character1: TMenuItem;
Left1: TMenuItem;
Right1: TMenuItem;
Center1: TMenuItem;
N4: TMenuItem;
Wordwrap1: TMenuItem;
N5: TMenuItem;
Font1: TMenuItem;
Editor: TRichEdit;
PopupMenu1: TPopupMenu;
Cut2: TMenuItem;
Copy2: TMenuItem;
Paste2: TMenuItem;
SaveFileDialog: TSaveDialog;
FontDialog1: TFontDialog;
Printersetup1: TMenuItem;
Close1: TMenuItem;
PrinterSetupDialog1: TPrinterSetupDialog;
PrintDialog1: TPrintDialog;
<------->procedure Exit1Click(Sender: TObject);
<------->procedure New1Click(Sender: TObject);

Left = 275
Top = 145
Width = 6
Height = 13
Caption = '0'
end
<-------> Object Label4: TLabel
Left = 81
Top = 167
Width = 73
Height = 13
Caption = 'Memory in Use:'
end
<-------> Object FreeRes: TLabel
Left = 276
Top = 167
Width = 6
Height = 13
Caption = '0'
end
<-------> Object OKButton: TButton
Left = 269
Top = 208
Width = 75
Height = 25
Cancel = True
Caption = 'OK'
Default = True
ModalResult = 2
TabOrder = 0
end
End

Procedures54
الملف التنفيذي لبرنامج
(TextEdit)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١
unit MDIEdit;
interface

```
  EditForm: TEditForm;

const
  DefaultFileName = 'Untitled';

implementation

uses Clipbrd, Printers, MDIFrame;

{$R *.dfm}

<------->procedure
TEditForm.Exit1Click(Sender:
TObject);
begin
  FrameForm.Exit1Click(Sender);
end;

<------->procedure
TEditForm.New1Click(Sender:
TObject);
begin
  FrameForm.New1Click(Sender);
end;

<------->procedure
TEditForm.Open1Click(Sender:
TObject);
begin
  FrameForm.Open1Click(Sender);
end;

<------->procedure
TEditForm.AlignClick(Sender:
TObject);
begin
  Left1.Checked := False;
  Right1.Checked := False;
  Center1.Checked := False;
  with Sender as TMenuItem do
Checked := True;
  with Editor.Paragraph do
    if Left1.Checked then
      Alignment := taLeftJustify
    else if Right1.Checked then
      Alignment := taRightJustify
    else if Center1.Checked then
      Alignment := taCenter;
```

```
    <------->procedure
AlignClick(Sender: TObject);
    <------->procedure
Wordwrap1Click(Sender: TObject);
    <------->procedure
Cut1Click(Sender: TObject);
    <------->procedure
Copy1Click(Sender: TObject);
    <------->procedure
Paste1Click(Sender: TObject);
    <------->procedure
Selectall1Click(Sender: TObject);
    <------->procedure
Delete1Click(Sender: TObject);
    <------->procedure
Edit1Click(Sender: TObject);
    <------->procedure
Saveas1Click(Sender: TObject);
    <------->procedure
Save1Click(Sender: TObject);
    <------->procedure
Font1Click(Sender: TObject);
    <------->procedure
Close1Click(Sender: TObject);
    <------->procedure
FormClose(Sender: TObject; var
Action: TCloseAction);
    <------->procedure
FormCloseQuery(Sender: TObject;
var CanClose: Boolean);
    <------->procedure
FormCreate(Sender: TObject);
    <------->procedure
Printersetup1Click(Sender:
TObject);
    <------->procedure
Print1Click(Sender: TObject);
    <------->procedure
Open1Click(Sender: TObject);
  private
    { Private declarations }
    PathName: string;
  public
    { Public declarations }
    <------->procedure Open(const
AFileName: string);
  end;

var
```

```
TObject);
begin
  Editor.ClearSelection;
end;

<------->procedure
TEditForm.Edit1Click(Sender:
TObject);
var
  HasSelection: Boolean;
begin
  Paste1.Enabled :=
Clipboard.HasFormat(CF_TEXT);
  Paste2.Enabled := Paste1.Enabled;
  HasSelection := Editor.SelLength >
0;
  Cut1.Enabled := HasSelection;
  Cut2.Enabled := HasSelection;
  Copy1.Enabled := HasSelection;
  Copy2.Enabled := HasSelection;
  Delete1.Enabled := HasSelection;
end;

<------->procedure
TEditForm.Open(const AFileName:
string);
begin
  PathName := AFileName;
  Caption :=
ExtractFileName(AFileName);
  with Editor do
  begin
    Lines.LoadFromFile(PathName);
    SelStart := 0;
    Modified := False;
  end;
end;

<------->procedure
TEditForm.Saveas1Click(Sender:
TObject);
begin
  SaveFileDialog.FileName :=
PathName;
  if SaveFileDialog.Execute then
  begin
    PathName :=
SaveFileDialog.FileName;
    Caption :=
```

```
end;

<------->procedure
TEditForm.Wordwrap1Click(Sende
r: TObject);
begin
  with Editor do
  begin
    WordWrap := not WordWrap; {
toggle word wrapping }
    if WordWrap then
      ScrollBars := ssVertical
    else
      ScrollBars := ssBoth;
    WordWrap1.Checked :=
WordWrap; { set menu item check }
  end;
end;

<------->procedure
TEditForm.Cut1Click(Sender:
TObject);
begin
  Editor.CutToClipboard;
end;

<------->procedure
TEditForm.Copy1Click(Sender:
TObject);
begin
  Editor.CopyToClipboard;
end;

<------->procedure
TEditForm.Paste1Click(Sender:
TObject);
begin
  Editor.PasteFromClipboard;
end;

<------->procedure
TEditForm.Selectall1Click(Sender:
TObject);
begin
  Editor.SelectAll;
end;

<------->procedure
TEditForm.Delete1Click(Sender:
```

```
const
  SWarningText = 'Save changes to %s?';
begin
  if Editor.Modified then
  begin
    case MessageDlg(Format(SWarningText, [PathName]), mtConfirmation,
      [mbYes, mbNo, mbCancel], 0) of
      idYes: Save1Click(Self);
      idCancel: CanClose := False;
    end;
  end;
end;

<------->procedure TEditForm.FormCreate(Sender: TObject);
begin
  PathName := DefaultFileName;
end;

<------->procedure TEditForm.Printersetup1Click(Sender: TObject);
begin
  PrinterSetupDialog1.Execute;
end;

<------->procedure TEditForm.Print1Click(Sender: TObject);
begin
  if PrintDialog1.Execute then
    Editor.Print(PathName);
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

unit MDIFrame;

interface

uses
ExtractFileName(PathName);
    Save1Click(Sender);
  end;
end;

<------->procedure TEditForm.Save1Click(Sender: TObject);
begin
  if PathName = DefaultFileName then
    SaveAs1Click(Sender)
  else
  begin
    Editor.Lines.SaveToFile(PathName);
    Editor.Modified := False;
  end;
end;

<------->procedure TEditForm.Font1Click(Sender: TObject);
begin
  FontDialog1.Font := Editor.Font;
  if FontDialog1.Execute then
    Editor.SelAttributes.Assign(FontDialog1.Font);
end;

<------->procedure TEditForm.Close1Click(Sender: TObject);
begin
  Close;
end;

<------->procedure TEditForm.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  Action := caFree;
end;

<------->procedure TEditForm.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
```

```
  SysUtils, Windows, Messages,
Classes, Graphics, Controls, Forms,
Dialogs,
  Menus;

type
  TFrameForm = class(TForm)
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    New1: TMenuItem;
    Open1: TMenuItem;
    N1: TMenuItem;
    Exit1: TMenuItem;
    Window1: TMenuItem;
    Tile1: TMenuItem;
    Cascade1: TMenuItem;
    Arrangeicons1: TMenuItem;
    OpenFileDialog: TOpenDialog;
    <------->procedure
Exit1Click(Sender: TObject);
    <------->procedure
New1Click(Sender: TObject);
    <------->procedure
Tile1Click(Sender: TObject);
    <------->procedure
Cascade1Click(Sender: TObject);
    <------->procedure
Arrangeicons1Click(Sender:
TObject);
    <------->procedure
Open1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FrameForm: TFrameForm;

implementation

uses MDIEdit;

{$R *.dfm}

<------->procedure
TFrameForm.Exit1Click(Sender:
TObject);
begin
  Close;
end;

<------->procedure
TFrameForm.New1Click(Sender:
TObject);
begin
  TEditForm.Create(Self);
end;

<------->procedure
TFrameForm.Tile1Click(Sender:
TObject);
begin
  Tile;
end;

<------->procedure
TFrameForm.Cascade1Click(Sender
: TObject);
begin
  Cascade;
end;

<------->procedure
TFrameForm.Arrangeicons1Click(S
ender: TObject);
begin
  ArrangeIcons;
end;

<------->procedure
TFrameForm.Open1Click(Sender:
TObject);
begin
  if OpenFileDialog.Execute then
  with TEditForm.Create(Self) do
    Open(OpenFileDialog.FileName);
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\
```

Top = 248
<------->Object File1: TMenuItem
Caption = '&File'
<------->Object New1: TMenuItem
Caption = '&New'
OnClick = New1Click
end
<------->Object Open1: TMenuItem
Caption = '&Open...'
OnClick = Open1Click
end
<------->Object Close1: TMenuItem
Caption = '&Close'
OnClick = Close1Click
end
<------->Object Save1: TMenuItem
Caption = '&Save'
OnClick = Save1Click
end
<------->Object Saveas1: TMenuItem
Caption = 'Save &as...'
OnClick = Saveas1Click
end
<------->Object Print1: TMenuItem
Caption = '&Print'
OnClick = Print1Click
end
<------->Object Printersetup1: TMenuItem
Caption = 'Printer setup...'
OnClick = Printersetup1Click
end
<------->Object N2: TMenuItem
Caption = '-'
end
<------->Object Exit1: TMenuItem
Caption = 'E&xit'
OnClick = Exit1Click
end
end
<------->Object Edit1: TMenuItem

<------->Objects48
الملف لبرنامج
(TextEdit)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

<------->Object EditForm: TEditForm
Left = 196
Top = 112
Width = 435
Height = 308
Caption = 'Untitled'
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
FormStyle = fsMDIChild
Menu = MainMenu1
OldCreateOrder = True
PopupMenu = PopupMenu1
Position = poDefault
Visible = True
OnClose = FormClose
OnCloseQuery = FormCloseQuery
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<------->Object Editor: TRichEdit
Left = 0
Top = 0
Width = 427
Height = 254
Align = alClient
BorderStyle = bsNone
ScrollBars = ssVertical
TabOrder = 0
end
<------->Object MainMenu1: TMainMenu
Left = 16

```
TMenuItem
      Caption = '&Right'
      OnClick = AlignClick
    end
    <-------->Object Center1:
TMenuItem
      Caption = '&Center'
      OnClick = AlignClick
    end
    <-------->Object N4: TMenuItem
      Caption = '-'
    end
    <-------->Object Wordwrap1:
TMenuItem
      Caption = '&Word wrap'
      Checked = True
      OnClick = Wordwrap1Click
    end
    <-------->Object N5: TMenuItem
      Caption = '-'
    end
    <-------->Object Font1:
TMenuItem
      Caption = '&Font...'
      OnClick = Font1Click
    end
   end
  end
  <-------->Object PopupMenu1:
TPopupMenu
   OnPopup = Edit1Click
   Left = 96
   Top = 64
   <-------->Object Cut2: TMenuItem
    Caption = 'Cu&t'
    OnClick = Cut1Click
   end
   <-------->Object Copy2:
TMenuItem
    Caption = '&Copy'
    OnClick = Copy1Click
   end
   <-------->Object Paste2:
TMenuItem
    Caption = '&Paste'
    OnClick = Paste1Click
   end
  end
```

```
    Caption = '&Edit'
    GroupIndex = 1
    OnClick = Edit1Click
    <-------->Object Cut1:
TMenuItem
      Caption = 'Cu&t'
      ShortCut = 16472
      OnClick = Cut1Click
    end
    <-------->Object Copy1:
TMenuItem
      Caption = '&Copy'
      ShortCut = 16451
      OnClick = Copy1Click
    end
    <-------->Object Paste1:
TMenuItem
      Caption = '&Paste'
      ShortCut = 16470
      OnClick = Paste1Click
    end
    <-------->Object Delete1:
TMenuItem
      Caption = 'De&lete'
      ShortCut = 16452
      OnClick = Delete1Click
    end
    <-------->Object N3: TMenuItem
      Caption = '-'
    end
    <-------->Object Selectall1:
TMenuItem
      Caption = 'Select &all'
      ShortCut = 16449
      OnClick = Selectall1Click
    end
   end
   <-------->Object Character1:
TMenuItem
    Caption = '&Character'
    GroupIndex = 1
    <-------->Object Left1:
TMenuItem
      Caption = '&Left'
      Checked = True
      OnClick = AlignClick
    end
    <-------->Object Right1:
```

```
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  FormStyle = fsMDIForm
  Menu = MainMenu1
  OldCreateOrder = True
  Position = poDefault
  WindowMenu = Window1
  PixelsPerInch = 96
  TextHeight = 13
  <------->Object MainMenu1: TMainMenu
    Left = 8
    Top = 248
    <------->Object File1: TMenuItem
      Caption = '&File'
      <------->Object New1: TMenuItem
        Caption = '&New'
        OnClick = New1Click
      end
      <------->Object Open1: TMenuItem
        Caption = '&Open...'
        OnClick = Open1Click
      end
      <------->Object N1: TMenuItem
        Caption = '-'
      end
      <------->Object Exit1: TMenuItem
        Caption = 'E&xit'
        GroupIndex = 9
        OnClick = Exit1Click
      end
    end
    <------->Object Window1: TMenuItem
      Caption = '&Window'
      GroupIndex = 9
      <------->Object Tile1: TMenuItem
        Caption = '&Tile'
        OnClick = Tile1Click
      end
      <------->Object Cascade1: TMenuItem
        Caption = '&Cascade'
```

```
  <------->Object SaveFileDialog: TSaveDialog
    Filter =
      'Rich text files (*.rtf)|*.rtf|Plain text files (*.txt)|*.txt|All' +
      ' files|*.*'
    Options = [ofHideReadOnly, ofFileMustExist, ofNoReadOnlyReturn]
    Left = 56
    Top = 64
  end
  <------->Object FontDialog1: TFontDialog
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    Left = 136
    Top = 64
  end
  <------->Object PrinterSetupDialog1: TPrinterSetupDialog
    Left = 176
    Top = 64
  end
  <------->Object PrintDialog1: TPrintDialog
    Left = 216
    Top = 64
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

<------->Object FrameForm: TFrameForm
  Left = -4
  Top = -4
  Width = 1032
  Height = 746
  Caption = 'Text editor'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
```

```
::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.no   mail: support@qusoft.no     ::
  ::                          fax: +47 22 41 74 91        ::

::::::::::::::::::::::::::::::::::::::::::::::::::::::::: }

unit Menu;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, Quickrpt, List, GrpList, manygrp, MD, ExtCtrls, qrextra;

type
  TMainForm = class(TForm)
    GroupBox1: TGroupBox;
    SimpleList: TRadioButton;
    GroupedList: TRadioButton;
    MasterDetail: TRadioButton;
    SQLMasterDetail: TRadioButton;
    Description: TMemo;
    Preview: TButton;
    Print: TButton;
    Exit: TButton;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Image2: TImage;
    RadioButton1: TRadioButton;
    CompositeReport: TQuickRep;
    QRCompositeReport1: TQRCompositeReport;
    <-------> procedure SimpleListClick(Sender: TObject);
    <-------> procedure PreviewClick(Sender: TObject);
    <-------> procedure
```

```
      OnClick = Cascade1Click
    end
    <------->Object Arrangeicons1: TMenuItem
      Caption = '&Arrange icons'
      OnClick = Arrangeicons1Click
    end
   end
  end
  <------->Object OpenFileDialog: TOpenDialog
   Filter =
     'Rich text files (*.rtf)|*.rtf|Plain text files (*.txt)|*.txt|All' +
     ' files|*.*'
   Left = 16
   Top = 24
  end
end
```

Procedures 22

الملف لبرنامج

(Qr2)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

برنامج لصناعة التقارير

QuickReport 2.0 for Delphi

الملف التنفيذي ويحتوي

```
{
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::
  :: QuickReport 2.0 for Delphi 1.0/2.0/3.0             ::
  ::
::
  :: Example reports project
::
  ::
::
  :: Copyright (c) 1997 QuSoft AS
```

```
begin
  if Report = CompositeReport then
  begin
    QRCompositeReport1.Preview;
  end else
    Report.Preview;
end;

<-------> procedure
TMainForm.PrintClick(Sender:
TObject);
begin
  if Report = CompositeReport then
  begin
    QRCompositeReport1.Print;
  end else
    Report.Print;
end;

<-------> procedure
TMainForm.ExitClick(Sender:
TObject);
begin
  Close;
end;

<-------> procedure
TMainForm.FormActivate(Sender:
TObject);
begin
  if Description.Lines.Count = 0 then
    SimpleListClick(Self);
end;

<-------> procedure
TMainForm.SimpleListClick(Sender
: TObject);
begin
  Report:=ListForm.QuickRep;
end;

<-------> procedure
TMainForm.GroupedListClick(Send
er: TObject);
begin
  Report := GrpListForm.QuickRep;
end;

```

```
PrintClick(Sender: TObject);
    <-------> procedure
ExitClick(Sender: TObject);
    <-------> procedure
FormActivate(Sender: TObject);
    <-------> procedure
GroupedListClick(Sender:
TObject);
    <-------> procedure
MasterDetailClick(Sender:
TObject);
    <-------> procedure
SQLMasterDetailClick(Sender:
TObject);
    <-------> procedure
RadioButton1Click(Sender:
TObject);
    <-------> procedure
QRCompositeReport1AddReports(S
ender: TObject);
  private
    FReport : TQuickRep;
    <-------> procedure
SetReport(Value : TQuickRep);
  public
    property Report : TQuickRep
read FReport write SetReport;
  end;

var
  MainForm: TMainForm;

implementation

{$R *.dfm}

<-------> procedure
TMainForm.SetReport(Value :
TQuickRep);
begin
  FReport:=Value;

Description.Lines.Assign(Report.Des
cription);
end;

<-------> procedure
TMainForm.PreviewClick(Sender:
TObject);
```

```
uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, quickrpt, Qrctrls,
DBTables, DB, ExtCtrls;

type
  TManyGrpForm = class(TForm)
    QuickRep: TQuickRep;
    RepQuery: TQuery;
    DetailBand1: TQRBand;
    QRGroup1: TQRGroup;
    QRGroup2: TQRGroup;
    QRDBText1: TQRDBText;
    QRDBText3: TQRDBText;
    QRBand1: TQRBand;
    QRBand2: TQRBand;
    QRDBText4: TQRDBText;
    QRExpr1: TQRExpr;
    QRDBText5: TQRDBText;
    QRExpr2: TQRExpr;
    QRExpr3: TQRExpr;
    RepQueryCustNo: TFloatField;
    RepQueryCompany:
TStringField;
    RepQueryAddr1: TStringField;
    RepQueryAddr2: TStringField;
    RepQueryCity: TStringField;
    RepQueryState: TStringField;
    RepQueryZip: TStringField;
    RepQueryCountry: TStringField;
    RepQueryPhone: TStringField;
    RepQueryFAX: TStringField;
    RepQueryTaxRate: TFloatField;
    RepQueryContact: TStringField;
    RepQueryLastInvoiceDate:
TDateTimeField;
    RepQueryOrderNo: TFloatField;
    RepQueryCustNo_1: TFloatField;
    RepQuerySaleDate:
TDateTimeField;
    RepQueryShipDate:
TDateTimeField;
    RepQueryEmpNo: TIntegerField;
    RepQueryShipToContact:
TStringField;
    RepQueryShipToAddr1:
```

```
<-------> procedure
TMainForm.MasterDetailClick(Sen
der: TObject);
begin
  Report:=MDForm.QuickRep;
end;

<-------> procedure
TMainForm.SQLMasterDetailClick(
Sender: TObject);
begin
  Report:=
ManyGrpForm.QuickRep;
end;

<-------> procedure
TMainForm.RadioButton1Click(Sen
der: TObject);
begin
  Report := CompositeReport;
end;

<-------> procedure
TMainForm.QRCompositeReport1
AddReports(Sender: TObject);
begin
  with QRCompositeReport1 do
  begin

Reports.Add(ListForm.QuickRep);

Reports.Add(GrpListForm.QuickRe
p);

Reports.Add(MDForm.QuickRep);

Reports.Add(ManyGrpForm.Quick
Rep);
  end;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
2
unit Manygrp;

interface

```

```
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  ManyGrpForm: TManyGrpForm;

implementation

{$R *.dfm}

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
2
unit Md;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, ExtCtrls, quickrpt, DB, DBTables, Qrctrls;

type
  TMDForm = class(TForm)
    QuickRep: TQuickRep;
    DetailBand1: TQRBand;
    ColumnHeaderBand1: TQRBand;
    PageFooterBand1: TQRBand;
    TitleBand1: TQRBand;
    Customer: TTable;
    QRDBText1: TQRDBText;
    QRLabel1: TQRLabel;
    QRDBText2: TQRDBText;
    QRLabel2: TQRLabel;
    QRDBText3: TQRDBText;
    QRLabel3: TQRLabel;
    QRDBText4: TQRDBText;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRDBText5: TQRDBText;
    QRLabel6: TQRLabel;
    QRSysData1: TQRSysData;
    QRSysData2: TQRSysData;
```

```
TStringField;
    RepQueryShipToAddr2: TStringField;
    RepQueryShipToCity: TStringField;
    RepQueryShipToState: TStringField;
    RepQueryShipToZip: TStringField;
    RepQueryShipToCountry: TStringField;
    RepQueryShipToPhone: TStringField;
    RepQueryShipVIA: TStringField;
    RepQueryPO: TStringField;
    RepQueryTerms: TStringField;
    RepQueryPaymentMethod: TStringField;
    RepQueryItemsTotal: TCurrencyField;
    RepQueryTaxRate_1: TFloatField;
    RepQueryFreight: TCurrencyField;
    RepQueryAmountPaid: TCurrencyField;
    RepQueryOrderNo_1: TFloatField;
    RepQueryItemNo: TFloatField;
    RepQueryPartNo: TFloatField;
    RepQueryQty: TIntegerField;
    RepQueryDiscount: TFloatField;
    RepQueryPartNo_1: TFloatField;
    RepQueryVendorNo: TFloatField;
    RepQueryDescription: TStringField;
    RepQueryOnHand: TFloatField;
    RepQueryOnOrder: TFloatField;
    RepQueryCost: TCurrencyField;
    RepQueryListPrice: TCurrencyField;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRExpr4: TQRExpr;
    QRLabel5: TQRLabel;
    ChildBand1: TQRChildBand;
```

```
    TitleBand1: TQRBand;
    Customer: TTable;
    QRDBText1: TQRDBText;
    QRLabel1: TQRLabel;
    QRDBText2: TQRDBText;
    QRLabel2: TQRLabel;
    QRDBText3: TQRDBText;
    QRLabel3: TQRLabel;
    QRDBText4: TQRDBText;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRDBText5: TQRDBText;
    QRLabel6: TQRLabel;
    QRSysData1: TQRSysData;
    QRSysData2: TQRSysData;
    QRExpr1: TQRExpr;
    QRGroup1: TQRGroup;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  GrpListForm: TGrpListForm;

implementation

{$R *.dfm}

end.
\\\\\\\\\\\\\\\\\\\\\\
5
unit List;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, ExtCtrls, quickrpt,
DB, DBTables, Qrctrls;

type
  TListForm = class(TForm)
    QuickRep: TQuickRep;
    DetailBand1: TQRBand;
```

```
    Orders: TTable;
    CustomerDS: TDataSource;
    QRSubDetail1: TQRSubDetail;
    QRDBText6: TQRDBText;
    QRDBText7: TQRDBText;
    QRDBText10: TQRDBText;
    QRLabel7: TQRLabel;
    QRLabel8: TQRLabel;
    QRLabel9: TQRLabel;
    GroupFooterBand1: TQRBand;
    QRExpr1: TQRExpr;
    QRLabel10: TQRLabel;
    QRLabel11: TQRLabel;
    QRExpr2: TQRExpr;
    ChildBand1: TQRChildBand;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  MDForm: TMDForm;

implementation

{$R *.dfm}

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
4
unit Grplist;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, ExtCtrls, quickrpt,
DB, DBTables, Qrctrls;

type
  TGrpListForm = class(TForm)
    QuickRep: TQuickRep;
    DetailBand1: TQRBand;
    ColumnHeaderBand1: TQRBand;
    PageFooterBand1: TQRBand;
```

```
  Top = 89
  Width = 787
  Height = 618
  Caption = 'MDForm'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'System'
  Font.Style = []
  OldCreateOrder = True
  Scaled = False
  PixelsPerInch = 96
  TextHeight = 16
  <------> Object Customer: TTable
    Active = True
    DatabaseName = 'DBDEMOS'
    IndexFieldNames = 'Company'
    ReadOnly = True
    TableName = 'CUSTOMER.DB'
    Left = 20
    Top = 76
  end
  <------> Object Orders: TTable
    Active = True
    DatabaseName = 'DBDEMOS'
    IndexFieldNames = 'CustNo'
    MasterFields = 'CustNo'
    MasterSource = CustomerDS
    ReadOnly = True
    TableName = 'ORDERS.DB'
    Left = 12
    Top = 132
  end
  <------> Object CustomerDS:
TDataSource
    DataSet = Customer
    Left = 28
    Top = 20
  end
End
\\\\\\\\\\\\\\\\\\\\\\
2
<------> Object ManyGrpForm:
TManyGrpForm
  Left = 135
  Top = 87
```

```
    ColumnHeaderBand1: TQRBand;
    PageFooterBand1: TQRBand;
    TitleBand1: TQRBand;
    Customer: TTable;
    QRDBText1: TQRDBText;
    QRLabel1: TQRLabel;
    QRDBText2: TQRDBText;
    QRLabel2: TQRLabel;
    QRDBText3: TQRDBText;
    QRLabel3: TQRLabel;
    QRDBText4: TQRDBText;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRDBText5: TQRDBText;
    QRLabel6: TQRLabel;
    QRSysData1: TQRSysData;
    QRSysData2: TQRSysData;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  ListForm: TListForm;

implementation

{$R *.dfm}

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

```
<------> ObjectS72
```

الملف النصي لبرنامج

(Qr2)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
1
<------> Object MDForm:
TMDForm
  Left = 117
```

```
    Size = 15
  end
  <------> Object RepQueryState:
TStringField
    FieldName = 'State'
  end
  <------> Object RepQueryZip:
TStringField
    FieldName = 'Zip'
    Size = 10
  end
  <------> Object
RepQueryCountry: TStringField
    FieldName = 'Country'
  end
  <------> Object RepQueryPhone:
TStringField
    FieldName = 'Phone'
    Size = 15
  end
  <------> Object RepQueryFAX:
TStringField
    FieldName = 'FAX'
    Size = 15
  end
  <------> Object
RepQueryTaxRate: TFloatField
    FieldName = 'TaxRate'
  end
  <------> Object RepQueryContact:
TStringField
    FieldName = 'Contact'
  end
  <------> Object
RepQueryLastInvoiceDate:
TDateTimeField
    FieldName = 'LastInvoiceDate'
  end
  <------> Object
RepQueryOrderNo: TFloatField
    FieldName = 'OrderNo'
  end
  <------> Object
RepQueryCustNo_1: TFloatField
    FieldName = 'CustNo_1'
  end
  <------> Object
RepQuerySaleDate:
TDateTimeField
```

```
  Width = 770
  Height = 657
  Caption = 'ManyGrpForm'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'System'
  Font.Style = []
  OldCreateOrder = True
  Scaled = False
  PixelsPerInch = 96
  TextHeight = 16
  <------> Object RepQuery: TQuery
    Active = True
    DatabaseName = 'DBDEMOS'
    SQL.Strings = (
      'select * from customer a, orders
b, items c, parts d'
      'where a.custno = b.custno'
      '  and b.orderno = c.orderno'
      '  and c.partno = d.partno'
      'order by a.company, orderno')
    Left = 18
    Top = 10
    <------> Object RepQueryCustNo:
TFloatField
      FieldName = 'CustNo'
    end
    <------> Object
RepQueryCompany: TStringField
      FieldName = 'Company'
      Size = 30
    end
    <------> Object RepQueryAddr1:
TStringField
      FieldName = 'Addr1'
      Size = 30
    end
    <------> Object RepQueryAddr2:
TStringField
      FieldName = 'Addr2'
      Size = 30
    end
    <------> Object RepQueryCity:
TStringField
      FieldName = 'City'
```

```
TStringField
    FieldName = 'ShipToPhone'
    Size = 15
  end
  <------> Object RepQueryShipVIA: TStringField
    FieldName = 'ShipVIA'
    Size = 7
  end
  <------> Object RepQueryPO: TStringField
    FieldName = 'PO'
    Size = 15
  end
  <------> Object RepQueryTerms: TStringField
    FieldName = 'Terms'
    Size = 6
  end
  <------> Object RepQueryPaymentMethod: TStringField
    FieldName = 'PaymentMethod'
    Size = 7
  end
  <------> Object RepQueryItemsTotal: TCurrencyField
    FieldName = 'ItemsTotal'
  end
  <------> Object RepQueryTaxRate_1: TFloatField
    FieldName = 'TaxRate_1'
  end
  <------> Object RepQueryFreight: TCurrencyField
    FieldName = 'Freight'
  end
  <------> Object RepQueryAmountPaid: TCurrencyField
    FieldName = 'AmountPaid'
  end
  <------> Object RepQueryOrderNo_1: TFloatField
    FieldName = 'OrderNo_1'
  end
  <------> Object RepQueryItemNo: TFloatField
```

```
    FieldName = 'SaleDate'
  end
  <------> Object RepQueryShipDate: TDateTimeField
    FieldName = 'ShipDate'
  end
  <------> Object RepQueryEmpNo: TIntegerField
    FieldName = 'EmpNo'
  end
  <------> Object RepQueryShipToContact: TStringField
    FieldName = 'ShipToContact'
  end
  <------> Object RepQueryShipToAddr1: TStringField
    FieldName = 'ShipToAddr1'
    Size = 30
  end
  <------> Object RepQueryShipToAddr2: TStringField
    FieldName = 'ShipToAddr2'
    Size = 30
  end
  <------> Object RepQueryShipToCity: TStringField
    FieldName = 'ShipToCity'
    Size = 15
  end
  <------> Object RepQueryShipToState: TStringField
    FieldName = 'ShipToState'
  end
  <------> Object RepQueryShipToZip: TStringField
    FieldName = 'ShipToZip'
    Size = 10
  end
  <------> Object RepQueryShipToCountry: TStringField
    FieldName = 'ShipToCountry'
  end
  <------> Object RepQueryShipToPhone:
```

```
TMainForm
  Left = 182
  Top = 121
  BorderStyle = bsDialog
  Caption = 'QuickReport 2.0 Example reports'
  ClientHeight = 331
  ClientWidth = 552
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  Scaled = False
  OnActivate = FormActivate
  PixelsPerInch = 96
  TextHeight = 16
  <------> Object Label1: TLabel
    Left = 56
    Top = 4
    Width = 174
    Height = 34
    Caption = 'QuickReport'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -29
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    Transparent = True
  end
  <------> Object Label2: TLabel
    Left = 236
    Top = 18
    Width = 70
    Height = 16
    Caption = 'Version 2.0'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'Arial'
    Font.Style = [fsBold]
```

```
      FieldName = 'ItemNo'
    end
    <------> Object RepQueryPartNo: TFloatField
      FieldName = 'PartNo'
    end
    <------> Object RepQueryQty: TIntegerField
      FieldName = 'Qty'
    end
    <------> Object RepQueryDiscount: TFloatField
      FieldName = 'Discount'
    end
    <------> Object RepQueryPartNo_1: TFloatField
      FieldName = 'PartNo_1'
    end
    <------> Object RepQueryVendorNo: TFloatField
      FieldName = 'VendorNo'
    end
    <------> Object RepQueryDescription: TStringField
      FieldName = 'Description'
      Size = 30
    end
    <------> Object RepQueryOnHand: TFloatField
      FieldName = 'OnHand'
    end
    <------> Object RepQueryOnOrder: TFloatField
      FieldName = 'OnOrder'
    end
    <------> Object RepQueryCost: TCurrencyField
      FieldName = 'Cost'
    end
    <------> Object RepQueryListPrice: TCurrencyField
      FieldName = 'ListPrice'
    end
  end
End
\\\\\\\\\\\\\\\\\\\\
3
<------> Object MainForm:
```

Width = 189
Height = 17
Caption = 'Simple list with group'
TabOrder = 1
OnClick = GroupedListClick
end
<------> Object MasterDetail: TRadioButton
Left = 12
Top = 67
Width = 193
Height = 17
Caption = 'Master/detail'
TabOrder = 2
OnClick = MasterDetailClick
end
<------> Object SQLMasterDetail: TRadioButton
Left = 12
Top = 91
Width = 201
Height = 17
Caption = 'Several grouping levels'
TabOrder = 3
OnClick = SQLMasterDetailClick
end
<------> Object Description: TMemo
Left = 220
Top = 16
Width = 313
Height = 201
TabStop = False
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
ReadOnly = True
TabOrder = 4
end
<------> Object RadioButton1:

ParentFont = False
Transparent = True
end
<------> Object Label3: TLabel
Left = 56
Top = 36
Width = 333
Height = 16
Caption = 'Copyright 1995 - 1997 QuSoft AS http://www.qusoft.com'
Transparent = True
end
<------> Object Image2: TImage
Left = 6
Top = 6
Width = 47
Height = 47
Picture.Data = {
C0C0C0C0C0C0C0C0C0C0C0C0C 0C0}
end
<------> Object GroupBox1: TGroupBox
Left = 4
Top = 58
Width = 541
Height = 225
Caption = 'Select report'
Color = clBtnFace
ParentColor = False
TabOrder = 0
<------> Object SimpleList: TRadioButton
Left = 12
Top = 20
Width = 193
Height = 17
Caption = 'Simple list'
Checked = True
TabOrder = 0
TabStop = True
OnClick = SimpleListClick
end
<------> Object GroupedList: TRadioButton
Left = 12
Top = 44

Caption = 'ListForm'
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'System'
Font.Style = []
OldCreateOrder = True
Scaled = False
PixelsPerInch = 96
TextHeight = 16
<------> Object Customer: TTable
Active = True
DatabaseName = 'DBDEMOS'
IndexFieldNames = 'Company'
ReadOnly = True
TableName = 'CUSTOMER.DB'
Left = 4
Top = 4
end
End
\\
5
<------> Object GrpListForm: TGrpListForm
Left = 138
Top = 108
Width = 680
Height = 587
Caption = 'GrpListForm'
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'System'
Font.Style = []
OldCreateOrder = True
Scaled = False
PixelsPerInch = 96
TextHeight = 16
<------> Object Customer: TTable
Active = True
DatabaseName = 'DBDEMOS'
IndexFieldNames = 'Company'
ReadOnly = True

TRadioButton
Left = 12
Top = 114
Width = 179
Height = 17
Caption = 'Composite report'
TabOrder = 5
OnClick = RadioButton1Click
end
end
<------> Object Preview: TButton
Left = 4
Top = 292
Width = 89
Height = 29
Caption = 'P&review'
TabOrder = 1
OnClick = PreviewClick
end
<------> Object Print: TButton
Left = 100
Top = 292
Width = 89
Height = 29
Caption = '&Print'
TabOrder = 2
OnClick = PrintClick
end
<------> Object Exit: TButton
Left = 456
Top = 292
Width = 89
Height = 29
Caption = 'E&xit'
TabOrder = 3
OnClick = ExitClick
end
End
\\\\\\\\\\\\\\\\\\\\\
5
<------> Object ListForm: TListForm
Left = 138
Top = 115
Width = 685
Height = 587

```
  ExtCtrls, StdCtrls, quickrpt, Db,
DBTables, printers, qrextra,
QRExport,
  qrprntr;

const
 Composite_Description = 'The
composite control can be used to link
several'+
                        ' reports together as a
single report.';

type
  TfrmQR3Demo = class(TForm)
    QRTextFilter1: TQRTextFilter;
    QRCSVFilter1: TQRCSVFilter;
    QRHTMLFilter1:
TQRHTMLFilter;
    QRCompositeReport1:
TQRCompositeReport;
    Label1: TLabel;
    VersionLbl: TLabel;
    Label3: TLabel;
    GroupBox1: TGroupBox;
    rbCreateList: TRadioButton;
    btnPreview: TButton;
    btnPrint: TButton;
    rbMasterDetail: TRadioButton;
    Description: TMemo;
    OpenDialog1: TOpenDialog;
    cbPreview: TComboBox;
    Label4: TLabel;
    rbExprMemo: TRadioButton;
    rbImage: TRadioButton;
    rbBasicMD: TRadioButton;
    btnFilters: TButton;
    rbComposite: TRadioButton;
    Label5: TLabel;
    rbNeedData: TRadioButton;
    rbFormLetter: TRadioButton;
    btnExport: TButton;
    SaveDialog1: TSaveDialog;
    rbAbout: TRadioButton;
    rbGrouping: TRadioButton;
    Image2: TImage;
    <--------> procedure
CreateList(Sender: TObject);
    <--------> procedure
```

```
    TableName = 'CUSTOMER.DB'
    Left = 12
    Top = 12
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

الملف التنفيذي لبرنامج

(Qr3)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
{ ::::::::::::::::::            علاء الدين اللباد
    ::::::::::::::::::::::::::::::
procedures 110
  :: QuickReport 3.0 for Delphi
3.0/4.0/5.0/6.0             ::
  ::
::
  :: Example reports project
::
  ::
::
  :: Copyright (c) 1995-1999 QuSoft
AS                    ::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.com  fax:
+47 22 41 74 91        ::

::::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::: }
unit mdmain;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
```

```
TCustomQuickRep read FReport write SetReport;
  end;

var
  frmQR3Demo: TfrmQR3Demo;

implementation

uses mdrpt, exprmemo, image, basicmd, needdata, frmltr,
  history, grouping;

{$R *.dfm}

<--------> procedure TfrmQR3Demo.SetReport(Value : TCustomQuickRep);
begin
  FReport := Value;
  if Value <> nil then
    if Value = TCustomQuickRep(QRCompositeReport1) then
      Description.Lines.Text := Composite_Description
    else
      Description.Lines.Assign(Report.Description);
end;

<--------> procedure TfrmQR3Demo.CreateList(Sender: TObject);
var
  aReport : TCustomQuickRep;
  SomeFields: TStringList;
  MyTable: TTable;
  nIdx: integer;
begin
  { Create a table on the fly, this example uses a table from the demo database }
  MyTable := TTable.Create(self);

  { create the list of fields to output from the table }
  SomeFields := TStringList.Create;
```

```
QRCompositeReport1AddReports(Sender: TObject);
    <--------> procedure btnCRClick(Sender: TObject);
    <--------> procedure rbCreateListClick(Sender: TObject);
    <--------> procedure rbMasterDetailClick(Sender: TObject);
    <--------> procedure FormCreate(Sender: TObject);
    <--------> procedure FormActivate(Sender: TObject);
    <--------> procedure btnPreviewClick(Sender: TObject);
    <--------> procedure rbExprMemoClick(Sender: TObject);
    <--------> procedure rbImageClick(Sender: TObject);
    <--------> procedure rbBasicMDClick(Sender: TObject);
    <--------> procedure btnFiltersClick(Sender: TObject);
    <--------> procedure rbCompositeClick(Sender: TObject);
    <--------> procedure rbNeedDataClick(Sender: TObject);
    <--------> procedure rbFormLetterClick(Sender: TObject);
    <--------> procedure btnExportClick(Sender: TObject);
    <--------> procedure rbAboutClick(Sender: TObject);
    <--------> procedure rbGroupingClick(Sender: TObject);
  private
    { Private declarations }
    FReport : TCustomQuickRep;
    CreateListReport : TQuickRep;
    <--------> procedure SetReport(Value : TCustomQuickRep);
  public
    { Public declarations }
    property Report :
```

```
  // report.

  for nIdx := 0 to
aReport.Bands.ColumnHeaderBand.
ControlCount -1 do
    if
aReport.Bands.ColumnHeaderBand.
Controls[nIdx] is TQRPrintable
then
      with
TQRPrintable(aReport.Bands.Colu
mnHeaderBand.Controls[nIdx]) do
        Left := Left - (5 * nIdx);

  for nIdx := 0 to
aReport.Bands.DetailBand.ControlC
ount -1 do
    if
aReport.Bands.DetailBand.Controls[
nIdx] is TQRPrintable then
      with
TQRPrintable(aReport.Bands.Detail
Band.Controls[nIdx]) do
        Left := Left - (5 * nIdx);


  { You can change the report objects
before calling the report }
//  areport.page.orientation :=
poLandscape;
  {preview or print the report}

  if sender = btnPreview then
  begin
    case cbPreview.ItemIndex of
      0: areport.preview;
      1: areport.previewModal;
      2: areport.previewModeless;
    end;
  end
  else if sender = btnPrint then
    areport.print;

  { all done, free the objects }
  aReport.Free;
  MyTable.Free;
  SomeFields.Free;
end;

```

```
  with MyTable do
  begin
    DatabaseName := 'DBDEMOS';
    TableName := 'COUNTRY.DB';
    ReadOnly := True;
    Active := True;

    // For this example, we will pull
the field names from the table
    // If you wanted to only use some
of the fields, you would edit
    // this list.
    for nIdx := 0 to FieldCount - 1 do

SomeFields.Add(Fields[nIdx].FieldN
ame);
  end;

  // If you didn't create the report,
you must set the report object to nil
  // before calling QRCreateList

  areport := nil;

  { Build the report }

  // If you change the displaywidth, it
will be reflecte in the created
  // report

  with MyTable.Fields[1] do
    DisplayWidth := DisplayWidth
div 2;

  // create the report
  QRCreateList(aReport, nil,
MyTable, 'Country Listing',
SomeFields);

  // Make the column header's font
use bold attribute

aReport.Bands.ColumnHeaderBand.
Font.Style := [fsBold];

  // Now adjust the spacing of the
fields.  There isn't any reason to
  // do this, this is just to show how to
access the controls on the
```

```
  if i >=0 then
    VersionLbl.Caption := 'Version' + copy(cQRName, i, 99);
  // Create the report object used by the QRCreateList() function
  CreateListReport := TQuickRep.Create(self);
  CreateListReport.Description.Text := 'Example of how to call the QRCreateList function';
  cbPreview.ItemIndex := 0;
end;

<--------> procedure TfrmQR3Demo.FormActivate(Sender: TObject);
begin
  if Description.Lines.Count = 0 then
  begin
    rbCreateListClick(Self);
    rbCreateList.Checked := True;
  end;
end;

<--------> procedure TfrmQR3Demo.btnPreviewClick(Sender: TObject);
begin
  // This code is more complicated than what you would
  // typically need to run a report. Most of this code
  // is to handle the selection of the various types of
  // reports.

  if report <> nil then
  begin
    if report = CreateListReport then
      CreateList(Sender)
    else if report = TCustomQuickRep(QRCompositeReport1) then
    begin
      if sender = btnPreview then
        QRCompositeReport1.preview
      else
        QRCompositeReport1.print;
    end
```

```
<--------> procedure TfrmQR3Demo.QRCompositeReport1AddReports(Sender: TObject);
begin
  // The OnAddReports event is called by the CompositeReport
  // to add the reports to list of reports
  with QRCompositeReport1.Reports do
  begin
    Add(frmMasterDetail.QuickRep1);
    Add(frmBasicMD.QuickRep1);
    Add(frmImageRpt.QuickRep1);
  end;
end;

<--------> procedure TfrmQR3Demo.btnCRClick(Sender : TObject);
begin
  QRCompositeReport1.Preview;
end;

<--------> procedure TfrmQR3Demo.rbCreateListClick(Sender: TObject);
begin
  Report := CreateListReport;
end;

<--------> procedure TfrmQR3Demo.rbMasterDetailClick(Sender: TObject);
begin
  Report := frmMasterDetail.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.FormCreate(Sender : TObject);
var i: integer;
begin
  // Get the current QuickReport version number from
  // the qrprntr unit
  i := pos(' ', cQRName);
```

```
end;

<--------> procedure TfrmQR3Demo.rbCompositeClick(Sender: TObject);
begin
  Report := TCustomQuickRep(QRCompositeReport1);
end;

<--------> procedure TfrmQR3Demo.rbNeedDataClick(Sender: TObject);
begin
  Report := frmNeedData.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.rbFormLetterClick(Sender: TObject);
begin
  Report := frmFormLetter.QuickRep1;
end;

// The following code show how to explicitly call an export
// filter without going through the preview
<--------> procedure TfrmQR3Demo.btnExportClick(Sender: TObject);
begin
  btnExport.Enabled := False;
  with SaveDialog1 do
  begin
    if Execute then
    begin

frmFormLetter.QuickRep1.ExportToFilter(TQRCommaSeparatedFilter.Create(FileName));
{
Other filters:
HTML: TQRHTMLDocumentFilter
ASCII: TQRAsciiExportFilter
CSV: TQRCommaSeparatedFilter
```

```
   else
   begin
     if sender = btnPreview then
     begin
       case cbPreview.ItemIndex of
         0: report.preview;
         1: report.previewModal;
         2: report.previewModeless;
       end;
     end
     else if sender = btnPrint then
       report.print;
   end;
  end;
end;

<--------> procedure TfrmQR3Demo.rbExprMemoClick(Sender: TObject);
begin
  Report := frmExprmemo.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.rbImageClick(Sender: TObject);
begin
  Report := frmImageRpt.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.rbBasicMDClick(Sender: TObject);
begin
  Report := frmBasicMD.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.btnFiltersClick(Sender: TObject);
begin
  MessageDlg('When an export filter control is dropped on a form, it''s added to all of the previews',mtInformation, [mbok], 0);
```

```
::::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::: }
unit basicmd;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBTables, ExtCtrls, QuickRpt, Qrctrls;

type
  TfrmBasicMD = class(TForm)
    QuickRep1: TQuickRep;
    dsCustomer: TDataSource;
    tbCustomer: TTable;
    DetailBand1: TQRBand;
    QRSubDetail1: TQRSubDetail;
    QRDBText1: TQRDBText;
    QRDBText2: TQRDBText;
    qryOrders: TQuery;
    QRGroup1: TQRGroup;
    QRDBText3: TQRDBText;
    ColumnHeaderBand1: TQRBand;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRLabel3: TQRLabel;
    QRDBText4: TQRDBText;
    QRLabel4: TQRLabel;
    QRBand1: TQRBand;
    QRExpr1: TQRExpr;
    QRLabel5: TQRLabel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmBasicMD: TfrmBasicMD;

implementation

{$R *.dfm}

```

```

In Professional Version:
RTF: TQRRTFExportFilter
WMF: TQRWMFExportFilter
Excel: TQRXLSFilter
}
    end;
  end;
  btnExport.Enabled := True;
end;

<--------> procedure TfrmQR3Demo.rbAboutClick(Sender: TObject);
begin
  Report := frmHistory.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.rbGroupingClick(Sender: TObject);
begin
  Report := frmGrouping.QuickRep1;
end;

end.

\\\\\\\\\\\\\\\\\\\\
2
{
::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::::
  :: QuickReport 3.0 for Delphi 3.0/4.0/5.0                ::
  ::                                                        ::
  :: Basic Master/Detail report with group bands           ::
  ::                                                        ::
  :: Copyright (c) 1995-1999 QuSoft AS                     ::
  :: All Rights Reserved                                    ::
  ::                                                        ::
  :: web: http://www.qusoft.com  fax: +47 22 41 74 91      ::
```

```
    tbCustomerCity: TStringField;
    tbCustomerState: TStringField;
    tbCustomerZip: TStringField;
    tbCustomerCountry: TStringField;
    tbCustomerPhone: TStringField;
    tbCustomerFAX: TStringField;
    tbCustomerTaxRate: TFloatField;
    tbCustomerContact: TStringField;
    tbCustomerLastInvoiceDate: TDateTimeField;
    QRExprMemo1: TQRExprMemo;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmExprmemo: TfrmExprmemo;

implementation

{$R *.dfm}

end.
\\\\\\\\\\\\\\\\
4
{
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::
  :: QuickReport 3.0 for Delphi 3.0/4.0/5.0                ::
  ::                                                        ::
  :: Simple report for doing a form letter                  ::
  ::                                                        ::
  :: Copyright (c) 1995-1999 QuSoft AS                      ::
  :: All Rights Reserved                                    ::
  ::                                                        ::
  :: web: http://www.qusoft.com  fax: +47 22 41 74 91       ::
```

```
end.

\\\\\\\

\\\\\\\\\\\\\\\\\\\\\\\
3
{
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::
  :: QuickReport 3.0 for Delphi 3.0/4.0/5.0                ::
  ::                                                        ::
  :: Demonstration on how to use the TQRExprMemo control    ::
  ::                                                        ::
  :: Copyright (c) 1995-1999 QuSoft AS                      ::
  :: All Rights Reserved                                    ::
  ::                                                        ::
  :: web: http://www.qusoft.com  fax: +47 22 41 74 91       ::

::::::::::::::::::::::::::::::::::::::::::::::::::::::::: }
unit exprmemo;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  QuickRpt, Qrctrls, Db, DBTables, ExtCtrls;

type
  TfrmExprmemo = class(TForm)
    QuickRep1: TQuickRep;
    tbCustomer: TTable;
    DetailBand1: TQRBand;
    tbCustomerCustNo: TFloatField;
    tbCustomerCompany: TStringField;
    tbCustomerAddr1: TStringField;
    tbCustomerAddr2: TStringField;
```

```
:::::::::::::::::::::::::::::::::::::::::::::
::::::::::::: }
unit frmltr;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  Db, DBTables, ExtCtrls, QuickRpt,
Qrctrls;

type
  TfrmFormLetter = class(TForm)
    QuickRep1: TQuickRep;
    qryEmployee: TQuery;
    DetailBand1: TQRBand;
    qryEmployeeEmpNo:
TIntegerField;
    qryEmployeeLastName:
TStringField;
    qryEmployeeFirstName:
TStringField;
    qryEmployeePhoneExt:
TStringField;
    qryEmployeeHireDate:
TDateTimeField;
    qryEmployeeSalary: TFloatField;
    QRExprMemo1:
TQRExprMemo;
    PageHeaderBand1: TQRBand;
    QRSysData1: TQRSysData;
    <--------> procedure
DetailBand1BeforePrint(Sender:
TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure
QuickRep1BeforePrint(Sender:
TCustomQuickRep;
      var PrintReport: Boolean);
  private
    { Private declarations }
    FirstDetail: boolean;
  public
    { Public declarations }
  end;

var
  frmFormLetter: TfrmFormLetter;

implementation

{$R *.dfm}

<--------> procedure
TfrmFormLetter.DetailBand1Before
Print(Sender: TQRCustomBand;
  var PrintBand: Boolean);
begin
  if FirstDetail then
    FirstDetail := False
  else
    QuickRep1.NewPage;
end;

<--------> procedure
TfrmFormLetter.QuickRep1Before
Print(Sender: TCustomQuickRep;
  var PrintReport: Boolean);
begin
  FirstDetail := True;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
5

{
:::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::
  :: QuickReport 3.0 for Delphi
3.0/4.0/5.0                 ::
  ::
::
  :: Simple report for grouping data
::
  ::
::
  :: Copyright (c) 1995-1999 QuSoft
AS                     ::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.com  fax:
```

```
  QRLabel6: TQRLabel;
  QRLabel7: TQRLabel;
  QRLabel8: TQRLabel;
  PageHeaderBand1: TQRBand;
  QRSysData1: TQRSysData;
  QRSysData2: TQRSysData;
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
 frmGrouping: TfrmGrouping;

implementation

{$R *.dfm}

end.
\\\\\\\\\\\\\\\\\\\\\\\\

6

{
:::::::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::
 :: QuickReport 3.0 for Delphi
3.0/4.0/5.0            ::
 ::
::
 :: Simple report for print the
contents of a stringlist   ::
 ::
::
 :: Copyright (c) 1995-1999 QuSoft
AS               ::
 :: All Rights Reserved
::
 ::
::
 :: web: http://www.qusoft.com  fax:
+47 22 41 74 91       ::

:::::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::: }
unit history;

interface
+47 22 41 74 91       ::

:::::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::: }
unit grouping;

interface

uses
 Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
 Db, DBTables, ExtCtrls, QuickRpt,
Qrctrls;

type
 TfrmGrouping = class(TForm)
  QuickRep1: TQuickRep;
  MasterQry: TQuery;
  MasterQrycompany:
TStringField;
  MasterQryorderno: TFloatField;
  MasterQrycustno: TFloatField;
  MasterQryItemsTotal:
TCurrencyField;
  DetailBand1: TQRBand;
  QRDBText2: TQRDBText;
  QRGroup1: TQRGroup;
  QRDBText4: TQRDBText;
  DataSource1: TDataSource;
  DetailQry: TQuery;
  DetailQryOrderNo: TFloatField;
  DetailQryQty: TIntegerField;
  DetailQryItemNo: TFloatField;
  QRSubDetail1: TQRSubDetail;
  QRDBText1: TQRDBText;
  QRDBText5: TQRDBText;
  QRBand1: TQRBand;
  QRExpr1: TQRExpr;
  QRBand2: TQRBand;
  QRExpr2: TQRExpr;
  SummaryBand1: TQRBand;
  QRExpr3: TQRExpr;
  ColumnHeaderBand1: TQRBand;
  QRLabel1: TQRLabel;
  QRLabel2: TQRLabel;
  QRLabel4: TQRLabel;
  QRLabel5: TQRLabel;
```

```
+47 22 41 74 91        ::

:::::::::::::::::::::::::::::::::::::::::::::
::::::::::::: }
unit image;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  Qrctrls, QuickRpt, Db, DBTables,
ExtCtrls;

type
  TfrmImageRpt = class(TForm)
    QuickRep1: TQuickRep;
    tbBio: TTable;
    DetailBand1: TQRBand;
    tbBioSpeciesNo: TFloatField;
    tbBioCategory: TStringField;
    tbBioCommon_Name:
TStringField;
    tbBioSpeciesName: TStringField;
    tbBioLengthcm: TFloatField;
    tbBioLength_In: TFloatField;
    tbBioNotes: TMemoField;
    tbBioGraphic: TGraphicField;
    QRDBText1: TQRDBText;
    QRDBImage1: TQRDBImage;
    ColumnHeaderBand1: TQRBand;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmImageRpt: TfrmImageRpt;

implementation

{$R *.dfm}

end.
```

```

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  QuickRpt, Qrctrls, ExtCtrls;

type
  TfrmHistory = class(TForm)
    QuickRep1: TQuickRep;
    DetailBand1: TQRBand;
    QRMemo1: TQRMemo;
    PageHeaderBand1: TQRBand;
    QRSysData1: TQRSysData;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmHistory: TfrmHistory;

implementation

{$R *.dfm}

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
7
{
:::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::
  :: QuickReport 3.0 for Delphi
3.0/4.0/5.0                ::
  ::
::
  :: Simple report with graphics
::
  ::
::
  :: Copyright (c) 1995-1999 QuSoft
AS                      ::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.com  fax:
```

```
\\\\\\\\\\\\\\\\\\\\\
8

{
::::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::
  :: QuickReport 3.0 for Delphi
3.0/4.0/5.0              ::
  ::
::
  :: Master/Detail report with some
extra code            ::
  ::
::
  :: Copyright (c) 1995-1999 QuSoft
AS                ::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.com  fax:
+47 22 41 74 91       ::

::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::: }
unit mdrpt;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  QuickRpt, Qrctrls, Db, DBTables,
ExtCtrls;

type
  TfrmMasterDetail = class(TForm)
    QuickRep1: TQuickRep;
    DetailBand1: TQRBand;
    dsCustomer: TDataSource;
    qryCustomer: TQuery;
    PageHeaderBand1: TQRBand;
    QRDBText1: TQRDBText;
    QRSysData1: TQRSysData;
    QRSysData2: TQRSysData;
    QRSubDetailOrders:
TQRSubDetail;
```

```
    qryOrders: TQuery;
    QRDBText3: TQRDBText;
    QRDBText4: TQRDBText;
    dsOrders: TDataSource;
    qryItems: TQuery;
    QRSubDetailItems:
TQRSubDetail;
    QRDBText5: TQRDBText;
    QRDBText2: TQRDBText;
    QRDBText6: TQRDBText;
    QRDBText7: TQRDBText;
    QRDBText9: TQRDBText;
    QRGroupCust: TQRGroup;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRShapeGray: TQRShape;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRDBOrderNo: TQRDBText;
    QRDBSalesDate: TQRDBText;
    GroupFooterBand1: TQRBand;
    <--------> procedure
QRSubDetailOrdersBeforePrint(Sen
der: TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure
QRSubDetailItemsBeforePrint(Send
er: TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure
QRGroupCustBeforePrint(Sender:
TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure
DetailBand1BeforePrint(Sender:
TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure
QRSubDetailItemsAfterPrint(Sende
r: TQRCustomBand;
      BandPrinted: Boolean);
    <--------> procedure
PageHeaderBand1BeforePrint(Send
er: TQRCustomBand;
      var PrintBand: Boolean);
  private
    { Private declarations }
```

```
end;

<--------> procedure TfrmMasterDetail.QRSubDetailOrdersBeforePrint(
  Sender: TQRCustomBand; var PrintBand: Boolean);
begin
  // We are print the order information on the first item subdetail
  // record.  If there are no item records, then we print the subdetail

  // Enable the order fields on the item subdetail band.  After they are
  // printed once, they will be disabled until the next order/item set
  QRDBOrderNo.Enabled := true;
  QRDBSalesDate.Enabled := QRDBOrderNo.Enabled;

  // Only allow this band to print if there are no subdetails
  qryItems.First;
  PrintBand := qryItems.EOF;
end;

<--------> procedure TfrmMasterDetail.QRSubDetailItemsBeforePrint(
  Sender: TQRCustomBand; var PrintBand: Boolean);
begin
  // toggle the item background so that we can have alternating colors
  // like the greenbar paper we all know and love.
  with QRShapeGray.Brush do
    if Color = $00F0F0F0 then
      Color := $00E0E0E0
    else
      Color := $00F0F0F0;
end;

<--------> procedure TfrmMasterDetail.QRSubDetailItemsAfterPrint(
  Sender: TQRCustomBand;
```

```
  public
    { Public declarations }
  end;

var
  frmMasterDetail: TfrmMasterDetail;

implementation

{$R *.dfm}

<--------> procedure TfrmMasterDetail.DetailBand1BeforePrint(Sender: TQRCustomBand;
  var PrintBand: Boolean);
begin
  // If there are no subdetails, then we print this band.  Otherwise we
  // let the group header print the controls from this band.  Group bands
  // can be reprinted on page breaks, which detail and subdetails can't do.
  qryOrders.First;
  PrintBand := qryOrders.EOF;

  // If it's our turn to print and the group band has our controls, then
  // we take them back.
  if PrintBand and (Sender.ControlCount = 0) then
    with QRGroupCust do
      while ControlCount > 0 do
        Controls[0].Parent := Sender;
end;

<--------> procedure TfrmMasterDetail.QRGroupCustBeforePrint(Sender: TQRCustomBand;
  var PrintBand: Boolean);
begin
  // We grab the detail band fields right from under it.
  if Sender.ControlCount = 0 then
    with DetailBand1 do
      While ControlCount > 0 do
        Controls[0].Parent := Sender;
```

```
unit mdrpt;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  QuickRpt, Qrctrls, Db, DBTables, ExtCtrls;

type
  TfrmMasterDetail = class(TForm)
    QuickRep1: TQuickRep;
    DetailBand1: TQRBand;
    dsCustomer: TDataSource;
    qryCustomer: TQuery;
    PageHeaderBand1: TQRBand;
    QRDBText1: TQRDBText;
    QRSysData1: TQRSysData;
    QRSysData2: TQRSysData;
    QRSubDetailOrders: TQRSubDetail;
    qryOrders: TQuery;
    QRDBText3: TQRDBText;
    QRDBText4: TQRDBText;
    dsOrders: TDataSource;
    qryItems: TQuery;
    QRSubDetailItems: TQRSubDetail;
    QRDBText5: TQRDBText;
    QRDBText2: TQRDBText;
    QRDBText6: TQRDBText;
    QRDBText7: TQRDBText;
    QRDBText9: TQRDBText;
    QRGroupCust: TQRGroup;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRShapeGray: TQRShape;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRDBOrderNo: TQRDBText;
    QRDBSalesDate: TQRDBText;
    GroupFooterBand1: TQRBand;
    <--------> procedure QRSubDetailOrdersBeforePrint(Sen
BandPrinted: Boolean);
begin
  // After we print it once, we disable the controls
  QRDBOrderNo.Enabled := false;
  QRDBSalesDate.Enabled := QRDBOrderNo.Enabled;
end;

<--------> procedure TfrmMasterDetail.PageHeaderBand1BeforePrint(
  Sender: TQRCustomBand; var PrintBand: Boolean);
begin
  // Re-enable the order fields on the item subdetail so that the order
  // information will be repeated after a page break
  QRDBOrderNo.Enabled := true;
  QRDBSalesDate.Enabled := QRDBOrderNo.Enabled;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
{
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
  :: QuickReport 3.0 for Delphi 3.0/4.0/5.0          ::
  ::                                                  ::
  :: Master/Detail report with some extra code       ::
  ::                                                  ::
  :: Copyright (c) 1995-1999 QuSoft AS               ::
  :: All Rights Reserved                              ::
  ::                                                  ::
  :: web: http://www.qusoft.com  fax: +47 22 41 74 91  ::

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: }
```

```
  PrintBand := qryOrders.EOF;

  // If it's our turn to print and the group band has our controls, then
  // we take them back.
  if PrintBand and (Sender.ControlCount = 0) then
    with QRGroupCust do
      while ControlCount > 0 do
        Controls[0].Parent := Sender;
end;

<--------> procedure TfrmMasterDetail.QRGroupCustBeforePrint(Sender: TQRCustomBand;
  var PrintBand: Boolean);
begin
  // We grab the detail band fields right from under it.
  if Sender.ControlCount = 0 then
    with DetailBand1 do
      While ControlCount > 0 do
        Controls[0].Parent := Sender;
end;

<--------> procedure TfrmMasterDetail.QRSubDetailOrdersBeforePrint(
  Sender: TQRCustomBand; var PrintBand: Boolean);
begin
  // We are print the order information on the first item subdetail
  // record.  If there are no item records, then we print the subdetail

  // Enable the order fields on the item subdetail band.  After they are
  // printed once, they will be disabled until the next order/item set
  QRDBOrderNo.Enabled := true;
  QRDBSalesDate.Enabled := QRDBOrderNo.Enabled;

  // Only allow this band to print if there are no subdetails
  qryItems.First;
```

```
der: TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure QRSubDetailItemsBeforePrint(Sender: TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure QRGroupCustBeforePrint(Sender: TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure DetailBand1BeforePrint(Sender: TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure QRSubDetailItemsAfterPrint(Sender: TQRCustomBand;
      BandPrinted: Boolean);
    <--------> procedure PageHeaderBand1BeforePrint(Sender: TQRCustomBand;
      var PrintBand: Boolean);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmMasterDetail: TfrmMasterDetail;

implementation

{$R *.dfm}

<--------> procedure TfrmMasterDetail.DetailBand1BeforePrint(Sender: TQRCustomBand;
  var PrintBand: Boolean);
begin
  // If there are no subdetails, then we print this band.  Otherwise we
  // let the group header print the controls from this band.  Group bands
  // can be reprinted on page breaks, which detail and subdetails can't do.
  qryOrders.First;
```

```
\\\\\\\\\\\\\\\\\\\\\\\\\\\\
9

{
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
  :: QuickReport 3.0 for Delphi 3.0/4.0/5.0                ::
  ::                                                       ::
  :: Demo report that is populated by the OnNeedData event   ::
  ::                                                       ::
  :: Copyright (c) 1995-1999 QuSoft AS                     ::
  :: All Rights Reserved                                   ::
  ::                                                       ::
  :: web: http://www.qusoft.com  fax: +47 22 41 74 91      ::

:::::::::::::::::::::::::::::::::::::::::::::::::::::::::: }
unit needdata;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, QuickRpt, Qrctrls;

type
  TfrmNeedData = class(TForm)
    QuickRep1: TQuickRep;
    DetailBand1: TQRBand;
    QRLabel1: TQRLabel;
    TitleBand1: TQRBand;
    QRSysData1: TQRSysData;
    <--------> procedure FormCreate(Sender: TObject);
    <--------> procedure QuickRep1BeforePrint(Sender: TCustomQuickRep;
      var PrintReport: Boolean);
    <--------> procedure
  PrintBand := qryItems.EOF;
end;

<--------> procedure TfrmMasterDetail.QRSubDetailItemsBeforePrint(
  Sender: TQRCustomBand; var PrintBand: Boolean);
begin
  // toggle the item background so that we can have alternating colors
  // like the greenbar paper we all know and love.
  with QRShapeGray.Brush do
    if Color = $00F0F0F0 then
      Color := $00E0E0E0
    else
      Color := $00F0F0F0;
end;

<--------> procedure TfrmMasterDetail.QRSubDetailItemsAfterPrint(
  Sender: TQRCustomBand; BandPrinted: Boolean);
begin
  // After we print it once, we disable the controls
  QRDBOrderNo.Enabled := false;
  QRDBSalesDate.Enabled := QRDBOrderNo.Enabled;
end;

<--------> procedure TfrmMasterDetail.PageHeaderBand1BeforePrint(
  Sender: TQRCustomBand; var PrintBand: Boolean);
begin
  // Re-enable the order fields on the item subdetail so that the order
  // information will be repeated after a page break
  QRDBOrderNo.Enabled := true;
  QRDBSalesDate.Enabled := QRDBOrderNo.Enabled;
end;

end.
```

```
  // If MoreData is true, then
QuickReport will print
  // another detail band.  When you
set it to false,
  // the report is done.

  MoreData := (CurrentIndex <
SomeList.Count);

  if MoreData then
  begin
    QRLabel1.Caption :=
SomeList[CurrentIndex];

    // Here's how to set the progress
bar
    QuickRep1.QRPrinter.Progress :=
(Longint(CurrentIndex) * 100) div
SomeList.Count;
  end
  else
    QuickRep1.QRPrinter.Progress :=
100;

  Inc(CurrentIndex);
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
10
{
::::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::
  :: QuickReport 3.0 for Delphi
3.0/4.0/5.0/6.0            ::
  ::
::
  :: Example reports project
::
  ::
::
  :: Copyright (c) 1995-1999 QuSoft
AS                    ::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.com  fax:
```

```
QuickRep1NeedData(Sender:
TObject; var MoreData: Boolean);
  private
    { Private declarations }
    SomeList: TStringlist;
    CurrentIndex: integer;
  public
    { Public declarations }
  end;

var
  frmNeedData: TfrmNeedData;

implementation

{$R *.dfm}

<--------> procedure
TfrmNeedData.FormCreate(Sender:
TObject);
var
  i: integer;
begin
  SomeList := TStringlist.Create;

  for i := 0 to 500 do
    SomeList.Add('Line ' +
IntToStr(i));
end;

<--------> procedure
TfrmNeedData.QuickRep1BeforePri
nt(Sender: TCustomQuickRep;
  var PrintReport: Boolean);
begin
  // You must reset your data in the
BeforePrint event
  // or when you print from the
preview, the report will
  // start with the last value(s)
  CurrentIndex := 0;
end;

<--------> procedure
TfrmNeedData.QuickRep1NeedData
(Sender: TObject;
  var MoreData: Boolean);
begin
```

```
+47 22 41 74 91        ::

::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: }
unit mdmain;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  ExtCtrls, StdCtrls, quickrpt, Db,
DBTables, printers, qrextra,
QRExport,
  qrprntr;

const
 Composite_Description = 'The
composite control can be used to link
several'+
                        ' reports together as a
single report.';

type
  TfrmQR3Demo = class(TForm)
    QRTextFilter1: TQRTextFilter;
    QRCSVFilter1: TQRCSVFilter;
    QRHTMLFilter1:
TQRHTMLFilter;
    QRCompositeReport1:
TQRCompositeReport;
    Label1: TLabel;
    VersionLbl: TLabel;
    Label3: TLabel;
    GroupBox1: TGroupBox;
    rbCreateList: TRadioButton;
    btnPreview: TButton;
    btnPrint: TButton;
    rbMasterDetail: TRadioButton;
    Description: TMemo;
    OpenDialog1: TOpenDialog;
    cbPreview: TComboBox;
    Label4: TLabel;
    rbExprMemo: TRadioButton;
    rbImage: TRadioButton;
    rbBasicMD: TRadioButton;
    btnFilters: TButton;
    rbComposite: TRadioButton;
    Label5: TLabel;
    rbNeedData: TRadioButton;
    rbFormLetter: TRadioButton;
    btnExport: TButton;
    SaveDialog1: TSaveDialog;
    rbAbout: TRadioButton;
    rbGrouping: TRadioButton;
    Image2: TImage;
    <--------> procedure
CreateList(Sender: TObject);
    <--------> procedure
QRCompositeReport1AddReports(S
ender: TObject);
    <--------> procedure
btnCRClick(Sender: TObject);
    <--------> procedure
rbCreateListClick(Sender:
TObject);
    <--------> procedure
rbMasterDetailClick(Sender:
TObject);
    <--------> procedure
FormCreate(Sender: TObject);
    <--------> procedure
FormActivate(Sender: TObject);
    <--------> procedure
btnPreviewClick(Sender: TObject);
    <--------> procedure
rbExprMemoClick(Sender:
TObject);
    <--------> procedure
rbImageClick(Sender: TObject);
    <--------> procedure
rbBasicMDClick(Sender: TObject);
    <--------> procedure
btnFiltersClick(Sender: TObject);
    <--------> procedure
rbCompositeClick(Sender:
TObject);
    <--------> procedure
rbNeedDataClick(Sender: TObject);
    <--------> procedure
rbFormLetterClick(Sender:
TObject);
    <--------> procedure
btnExportClick(Sender: TObject);
    <--------> procedure
rbAboutClick(Sender: TObject);
    <--------> procedure
```

```
  MyTable: TTable;
  nIdx: integer;
begin
  { Create a table on the fly, this example uses a table from the demo database }
  MyTable := TTable.Create(self);

  { create the list of fields to output from the table }
  SomeFields := TStringList.Create;
  with MyTable do
  begin
    DatabaseName := 'DBDEMOS';
    TableName := 'COUNTRY.DB';
    ReadOnly := True;
    Active := True;

    // For this example, we will pull the field names from the table
    // If you wanted to only use some of the fields, you would edit
    // this list.
    for nIdx := 0 to FieldCount - 1 do
      SomeFields.Add(Fields[nIdx].FieldName);
  end;

  // If you didn't create the report, you must set the report object to nil
  // before calling QRCreateList

  areport := nil;

  { Build the report }

  // If you change the displaywidth, it will be reflecte in the created
  // report

  with MyTable.Fields[1] do
    DisplayWidth := DisplayWidth div 2;

  // create the report
  QRCreateList(aReport, nil, MyTable, 'Country Listing', SomeFields);
rbGroupingClick(Sender: TObject);
  private
    { Private declarations }
    FReport : TCustomQuickRep;
    CreateListReport : TQuickRep;
    <--------> procedure SetReport(Value : TCustomQuickRep);
  public
    { Public declarations }
    property Report : TCustomQuickRep read FReport write SetReport;
  end;

var
  frmQR3Demo: TfrmQR3Demo;

implementation

uses mdrpt, exprmemo, image, basicmd, needdata, frmltr,
  history, grouping;

{$R *.dfm}

<--------> procedure TfrmQR3Demo.SetReport(Value : TCustomQuickRep);
begin
  FReport := Value;
  if Value <> nil then
    if Value = TCustomQuickRep(QRCompositeReport1) then
      Description.Lines.Text := Composite_Description
    else
      Description.Lines.Assign(Report.Description);
end;

<--------> procedure TfrmQR3Demo.CreateList(Sender: TObject);
var
  aReport : TCustomQuickRep;
  SomeFields: TStringList;
```

```
  end;
 end
 else if sender = btnPrint then
  areport.print;

 { all done, free the objects }
 aReport.Free;
 MyTable.Free;
 SomeFields.Free;
end;

<--------> procedure
TfrmQR3Demo.QRCompositeRepor
t1AddReports(Sender: TObject);
begin
 // The OnAddReports event is
called by the CompositeReport
 // to add the reports to list of
reports
 with
QRCompositeReport1.Reports do
 begin

Add(frmMasterDetail.QuickRep1);
  Add(frmBasicMD.QuickRep1);
  Add(frmImageRpt.QuickRep1);
 end;
end;

<--------> procedure
TfrmQR3Demo.btnCRClick(Sender
: TObject);
begin
 QRCompositeReport1.Preview;
end;

<--------> procedure
TfrmQR3Demo.rbCreateListClick(S
ender: TObject);
begin
 Report := CreateListReport;
end;

<--------> procedure
TfrmQR3Demo.rbMasterDetailClic
k(Sender: TObject);
begin
 Report :=
frmMasterDetail.QuickRep1;
```

```

 // Make the column header's font
use bold attribute

aReport.Bands.ColumnHeaderBand.
Font.Style := [fsBold];

 // Now adjust the spacing of the
fields.  There isn't any reason to
 // do this, this is just to show how to
access the controls on the
 // report.

 for nIdx := 0 to
aReport.Bands.ColumnHeaderBand.
ControlCount -1 do
  if
aReport.Bands.ColumnHeaderBand.
Controls[nIdx] is TQRPrintable
then
    with
TQRPrintable(aReport.Bands.Colu
mnHeaderBand.Controls[nIdx]) do
     Left := Left - (5 * nIdx);

 for nIdx := 0 to
aReport.Bands.DetailBand.ControlC
ount -1 do
  if
aReport.Bands.DetailBand.Controls[
nIdx] is TQRPrintable then
    with
TQRPrintable(aReport.Bands.Detail
Band.Controls[nIdx]) do
     Left := Left - (5 * nIdx);


 { You can change the report objects
before calling the report }
//  areport.page.orientation :=
poLandscape;
 {preview or print the report}

 if sender = btnPreview then
 begin
  case cbPreview.ItemIndex of
    0: areport.preview;
    1: areport.previewModal;
    2: areport.previewModeless;
```

```
  if report = CreateListReport then
    CreateList(Sender)
  else if report =
TCustomQuickRep(QRCompositeR
eport1) then
  begin
    if sender = btnPreview then
      QRCompositeReport1.preview
    else
      QRCompositeReport1.print;
  end
  else
  begin
    if sender = btnPreview then
    begin
      case cbPreview.ItemIndex of
        0: report.preview;
        1: report.previewModal;
        2: report.previewModeless;
      end;
    end
    else if sender = btnPrint then
      report.print;
  end;
 end;
end;

<--------> procedure
TfrmQR3Demo.rbExprMemoClick(
Sender: TObject);
begin
 Report :=
frmExprmemo.QuickRep1;
end;

<--------> procedure
TfrmQR3Demo.rbImageClick(Send
er: TObject);
begin
 Report :=
frmImageRpt.QuickRep1;
end;

<--------> procedure
TfrmQR3Demo.rbBasicMDClick(Se
nder: TObject);
begin
 Report :=
frmBasicMD.QuickRep1;
end;

<--------> procedure
TfrmQR3Demo.FormCreate(Sender
: TObject);
var i: integer;
begin
 // Get the current QuickReport
version number from
 // the qrprntr unit
 i := pos(' ', cQRName);
 if i >=0 then
   VersionLbl.Caption := 'Version' +
copy(cQRName, i, 99);
 // Create the report object used by
the QRCreateList() function
 CreateListReport :=
TQuickRep.Create(self);
 CreateListReport.Description.Text
:= 'Example of how to call the
QRCreateList function';
 cbPreview.ItemIndex := 0;
end;

<--------> procedure
TfrmQR3Demo.FormActivate(Send
er: TObject);
begin
 if Description.Lines.Count = 0 then
 begin
   rbCreateListClick(Self);
   rbCreateList.Checked := True;
 end;
end;

<--------> procedure
TfrmQR3Demo.btnPreviewClick(Se
nder: TObject);
begin
 // This code is more complicated
than what you would
 // typically need to run a report.
Most of this code
 // is to handle the selection of the
various types of
 // reports.

 if report <> nil then
 begin
```

```
    if Execute then
    begin

  frmFormLetter.QuickRep1.ExportToFilter(TQRCommaSeparatedFilter.Create(FileName));
{
Other filters:
HTML: TQRHTMLDocumentFilter
ASCII: TQRAsciiExportFilter
CSV: TQRCommaSeparatedFilter

In Professional Version:
RTF: TQRRTFExportFilter
WMF: TQRWMFExportFilter
Excel: TQRXLSFilter
}
    end;
  end;
  btnExport.Enabled := True;
end;

<--------> procedure TfrmQR3Demo.rbAboutClick(Sender: TObject);
begin
  Report := frmHistory.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.rbGroupingClick(Sender: TObject);
begin
  Report := frmGrouping.QuickRep1;
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
end;

<--------> procedure TfrmQR3Demo.btnFiltersClick(Sender: TObject);
begin
  MessageDlg('When an export filter control is dropped on a form, it''s added to all of the previews',mtInformation, [mbok], 0);
end;

<--------> procedure TfrmQR3Demo.rbCompositeClick(Sender: TObject);
begin
  Report := TCustomQuickRep(QRCompositeReport1);
end;

<--------> procedure TfrmQR3Demo.rbNeedDataClick(Sender: TObject);
begin
  Report := frmNeedData.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.rbFormLetterClick(Sender: TObject);
begin
  Report := frmFormLetter.QuickRep1;
end;

// The following code show how to explicitly call an export
// filter without going through the preview
<--------> procedure TfrmQR3Demo.btnExportClick(Sender: TObject);
begin
  btnExport.Enabled := False;
  with SaveDialog1 do
  begin
```

الملف التنفيذي لبرنامج
(Qr3)

إعداد
علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
{ ::::::::::::::::::            علاء الدين اللباد
   :::::::::::::::::::::::::::::
procedures 110
  :: QuickReport 3.0 for Delphi
3.0/4.0/5.0/6.0            ::
  ::
::
  :: Example reports project
::
  ::
::
  :: Copyright (c) 1995-1999 QuSoft
AS                    ::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.com  fax:
+47 22 41 74 91        ::

::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::: }
unit mdmain;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  ExtCtrls, StdCtrls, quickrpt, Db,
DBTables, printers, qrextra,
QRExport,
  qrprntr;

const
 Composite_Description = 'The
composite control can be used to link
several'+
                  ' reports together as a
single report.';

type
  TfrmQR3Demo = class(TForm)
    QRTextFilter1: TQRTextFilter;
    QRCSVFilter1: TQRCSVFilter;
    QRHTMLFilter1:
TQRHTMLFilter;
    QRCompositeReport1:
TQRCompositeReport;
    Label1: TLabel;
    VersionLbl: TLabel;
    Label3: TLabel;
    GroupBox1: TGroupBox;
    rbCreateList: TRadioButton;
    btnPreview: TButton;
    btnPrint: TButton;
    rbMasterDetail: TRadioButton;
    Description: TMemo;
    OpenDialog1: TOpenDialog;
    cbPreview: TComboBox;
    Label4: TLabel;
    rbExprMemo: TRadioButton;
    rbImage: TRadioButton;
    rbBasicMD: TRadioButton;
    btnFilters: TButton;
    rbComposite: TRadioButton;
    Label5: TLabel;
    rbNeedData: TRadioButton;
    rbFormLetter: TRadioButton;
    btnExport: TButton;
    SaveDialog1: TSaveDialog;
    rbAbout: TRadioButton;
    rbGrouping: TRadioButton;
    Image2: TImage;
    <--------> procedure
CreateList(Sender: TObject);
    <--------> procedure
QRCompositeReport1AddReports(S
ender: TObject);
    <--------> procedure
btnCRClick(Sender: TObject);
    <--------> procedure
rbCreateListClick(Sender:
TObject);
    <--------> procedure
rbMasterDetailClick(Sender:
TObject);
```

```
basicmd, needdata, frmltr,
  history, grouping;

{$R *.dfm}

<--------> procedure
TfrmQR3Demo.SetReport(Value :
TCustomQuickRep);
begin
  FReport := Value;
  if Value <> nil then
    if Value =
TCustomQuickRep(QRCompositeR
eport1) then
      Description.Lines.Text :=
Composite_Description
    else

Description.Lines.Assign(Report.Des
cription);
end;

<--------> procedure
TfrmQR3Demo.CreateList(Sender:
TObject);
var
  aReport : TCustomQuickRep;
  SomeFields: TStringList;
  MyTable: TTable;
  nIdx: integer;
begin
  { Create a table on the fly, this
example uses a table from the demo
database }
  MyTable := TTable.Create(self);

  { create the list of fields to output
from the table }
  SomeFields := TStringList.Create;
  with MyTable do
  begin
    DatabaseName := 'DBDEMOS';
    TableName := 'COUNTRY.DB';
    ReadOnly := True;
    Active := True;

    // For this example, we will pull
the field names from the table
    // If you wanted to only use some
```

```
    <--------> procedure
FormCreate(Sender: TObject);
    <--------> procedure
FormActivate(Sender: TObject);
    <--------> procedure
btnPreviewClick(Sender: TObject);
    <--------> procedure
rbExprMemoClick(Sender:
TObject);
    <--------> procedure
rbImageClick(Sender: TObject);
    <--------> procedure
rbBasicMDClick(Sender: TObject);
    <--------> procedure
btnFiltersClick(Sender: TObject);
    <--------> procedure
rbCompositeClick(Sender:
TObject);
    <--------> procedure
rbNeedDataClick(Sender: TObject);
    <--------> procedure
rbFormLetterClick(Sender:
TObject);
    <--------> procedure
btnExportClick(Sender: TObject);
    <--------> procedure
rbAboutClick(Sender: TObject);
    <--------> procedure
rbGroupingClick(Sender: TObject);
  private
    { Private declarations }
    FReport : TCustomQuickRep;
    CreateListReport : TQuickRep;
    <--------> procedure
SetReport(Value :
TCustomQuickRep);
  public
    { Public declarations }
    property Report :
TCustomQuickRep read FReport
write SetReport;
  end;

var
  frmQR3Demo: TfrmQR3Demo;

implementation

uses mdrpt, exprmemo, image,
```

```
TQRPrintable(aReport.Bands.ColumnHeaderBand.Controls[nIdx]) do
      Left := Left - (5 * nIdx);

  for nIdx := 0 to aReport.Bands.DetailBand.ControlCount -1 do
    if aReport.Bands.DetailBand.Controls[nIdx] is TQRPrintable then
      with TQRPrintable(aReport.Bands.DetailBand.Controls[nIdx]) do
        Left := Left - (5 * nIdx);


  { You can change the report objects before calling the report }
//  areport.page.orientation := poLandscape;
  {preview or print the report}

  if sender = btnPreview then
  begin
    case cbPreview.ItemIndex of
      0: areport.preview;
      1: areport.previewModal;
      2: areport.previewModeless;
    end;
  end
  else if sender = btnPrint then
    areport.print;

  { all done, free the objects }
  aReport.Free;
  MyTable.Free;
  SomeFields.Free;
end;

<--------> procedure TfrmQR3Demo.QRCompositeReport1AddReports(Sender: TObject);
begin
  // The OnAddReports event is called by the CompositeReport
  // to add the reports to list of reports
  with QRCompositeReport1.Reports do
of the fields, you would edit
    // this list.
    for nIdx := 0 to FieldCount - 1 do

SomeFields.Add(Fields[nIdx].FieldName);
  end;

  // If you didn't create the report, you must set the report object to nil
  // before calling QRCreateList

  areport := nil;

  { Build the report }

  // If you change the displaywidth, it will be reflecte in the created
  // report

  with MyTable.Fields[1] do
    DisplayWidth := DisplayWidth div 2;

  // create the report
  QRCreateList(aReport, nil, MyTable, 'Country Listing', SomeFields);

  // Make the column header's font use bold attribute

aReport.Bands.ColumnHeaderBand.Font.Style := [fsBold];

  // Now adjust the spacing of the fields.  There isn't any reason to
  // do this, this is just to show how to access the controls on the
  // report.

  for nIdx := 0 to aReport.Bands.ColumnHeaderBand.ControlCount -1 do
    if aReport.Bands.ColumnHeaderBand.Controls[nIdx] is TQRPrintable then
      with
```

```
  cbPreview.ItemIndex := 0;
end;

<--------> procedure
TfrmQR3Demo.FormActivate(Send
er: TObject);
begin
  if Description.Lines.Count = 0 then
  begin
    rbCreateListClick(Self);
    rbCreateList.Checked := True;
  end;
end;

<--------> procedure
TfrmQR3Demo.btnPreviewClick(Se
nder: TObject);
begin
  // This code is more complicated
than what you would
  // typically need to run a report.
Most of this code
  // is to handle the selection of the
various types of
  // reports.

  if report <> nil then
  begin
    if report = CreateListReport then
      CreateList(Sender)
    else if report =
TCustomQuickRep(QRCompositeR
eport1) then
    begin
      if sender = btnPreview then
        QRCompositeReport1.preview
      else
        QRCompositeReport1.print;
    end
    else
    begin
      if sender = btnPreview then
      begin
        case cbPreview.ItemIndex of
          0: report.preview;
          1: report.previewModal;
          2: report.previewModeless;
        end;
```

```
  begin

Add(frmMasterDetail.QuickRep1);
    Add(frmBasicMD.QuickRep1);
    Add(frmImageRpt.QuickRep1);
  end;
end;

<--------> procedure
TfrmQR3Demo.btnCRClick(Sender
: TObject);
begin
  QRCompositeReport1.Preview;
end;

<--------> procedure
TfrmQR3Demo.rbCreateListClick(S
ender: TObject);
begin
  Report := CreateListReport;
end;

<--------> procedure
TfrmQR3Demo.rbMasterDetailClic
k(Sender: TObject);
begin
  Report :=
frmMasterDetail.QuickRep1;
end;

<--------> procedure
TfrmQR3Demo.FormCreate(Sender
: TObject);
var i: integer;
begin
  // Get the current QuickReport
version number from
  // the qrprntr unit
  i := pos(' ', cQRName);
  if i >=0 then
    VersionLbl.Caption := 'Version' +
copy(cQRName, i, 99);
  // Create the report object used by
the QRCreateList() function
  CreateListReport :=
TQuickRep.Create(self);
  CreateListReport.Description.Text
:= 'Example of how to call the
QRCreateList function';
```

```
end;

<--------> procedure TfrmQR3Demo.rbNeedDataClick(Sender: TObject);
begin
  Report := frmNeedData.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.rbFormLetterClick(Sender: TObject);
begin
  Report := frmFormLetter.QuickRep1;
end;

// The following code show how to explicitly call an export
// filter without going through the preview
<--------> procedure TfrmQR3Demo.btnExportClick(Sender: TObject);
begin
  btnExport.Enabled := False;
  with SaveDialog1 do
  begin
    if Execute then
    begin
      frmFormLetter.QuickRep1.ExportToFilter(TQRCommaSeparatedFilter.Create(FileName));
{
Other filters:
HTML: TQRHTMLDocumentFilter
ASCII: TQRAsciiExportFilter
CSV: TQRCommaSeparatedFilter

In Professional Version:
RTF: TQRRTFExportFilter
WMF: TQRWMFExportFilter
Excel: TQRXLSFilter
}
    end;
  end;
  btnExport.Enabled := True;
    end
    else if sender = btnPrint then
      report.print;
    end;
  end;
end;

<--------> procedure TfrmQR3Demo.rbExprMemoClick(Sender: TObject);
begin
  Report := frmExprmemo.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.rbImageClick(Sender: TObject);
begin
  Report := frmImageRpt.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.rbBasicMDClick(Sender: TObject);
begin
  Report := frmBasicMD.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.btnFiltersClick(Sender: TObject);
begin
  MessageDlg('When an export filter control is dropped on a form, it''s added to all of the previews',mtInformation, [mbok], 0);
end;

<--------> procedure TfrmQR3Demo.rbCompositeClick(Sender: TObject);
begin
  Report := TCustomQuickRep(QRCompositeReport1);
```

```
Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBTables, ExtCtrls, QuickRpt, Qrctrls;

type
  TfrmBasicMD = class(TForm)
    QuickRep1: TQuickRep;
    dsCustomer: TDataSource;
    tbCustomer: TTable;
    DetailBand1: TQRBand;
    QRSubDetail1: TQRSubDetail;
    QRDBText1: TQRDBText;
    QRDBText2: TQRDBText;
    qryOrders: TQuery;
    QRGroup1: TQRGroup;
    QRDBText3: TQRDBText;
    ColumnHeaderBand1: TQRBand;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRLabel3: TQRLabel;
    QRDBText4: TQRDBText;
    QRLabel4: TQRLabel;
    QRBand1: TQRBand;
    QRExpr1: TQRExpr;
    QRLabel5: TQRLabel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmBasicMD: TfrmBasicMD;

implementation

{$R *.dfm}

end.

\\\\\\\

\\\\\\\\\\\\\\\\\\\\\\
3
{
::::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::
```

```
end;

<--------> procedure
TfrmQR3Demo.rbAboutClick(Sender: TObject);
begin
  Report := frmHistory.QuickRep1;
end;

<--------> procedure
TfrmQR3Demo.rbGroupingClick(Sender: TObject);
begin
  Report :=
frmGrouping.QuickRep1;
end;

end.

\\\\\\\\\\\\\\\\\\\\
2
{
::::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::
  :: QuickReport 3.0 for Delphi
3.0/4.0/5.0                ::
  ::
::
  :: Basic Master/Detail report with
group bands            ::
  ::
::
  :: Copyright (c) 1995-1999 QuSoft
AS                     ::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.com  fax:
+47 22 41 74 91        ::

::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::: }
unit basicmd;

interface

uses
  Windows, Messages, SysUtils,
```

```
TStringField;
   tbCustomerLastInvoiceDate: TDateTimeField;
   QRExprMemo1: TQRExprMemo;
  private
   { Private declarations }
  public
   { Public declarations }
  end;

var
  frmExprmemo: TfrmExprmemo;

implementation

{$R *.dfm}

end.
\\\\\\\\\\\\\\\\
4
{
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
  :: QuickReport 3.0 for Delphi 3.0/4.0/5.0                 ::
  ::                                                         ::
  :: Simple report for doing a form letter                   ::
  ::                                                         ::
  :: Copyright (c) 1995-1999 QuSoft AS                       ::
  :: All Rights Reserved                                     ::
  ::                                                         ::
  :: web: http://www.qusoft.com  fax: +47 22 41 74 91        ::
::::::::::::::::::::::::::::::::::::::::::::::::::::::::: }
unit frmltr;

interface

uses
  Windows, Messages, SysUtils,
```

```
  :: QuickReport 3.0 for Delphi 3.0/4.0/5.0                 ::
  ::                                                         ::
  :: Demonstration on how to use the TQRExprMemo control     ::
  ::                                                         ::
  :: Copyright (c) 1995-1999 QuSoft AS                       ::
  :: All Rights Reserved                                     ::
  ::                                                         ::
  :: web: http://www.qusoft.com  fax: +47 22 41 74 91        ::
::::::::::::::::::::::::::::::::::::::::::::::::::::::::: }
unit exprmemo;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  QuickRpt, Qrctrls, Db, DBTables, ExtCtrls;

type
  TfrmExprmemo = class(TForm)
    QuickRep1: TQuickRep;
    tbCustomer: TTable;
    DetailBand1: TQRBand;
    tbCustomerCustNo: TFloatField;
    tbCustomerCompany: TStringField;
    tbCustomerAddr1: TStringField;
    tbCustomerAddr2: TStringField;
    tbCustomerCity: TStringField;
    tbCustomerState: TStringField;
    tbCustomerZip: TStringField;
    tbCustomerCountry: TStringField;
    tbCustomerPhone: TStringField;
    tbCustomerFAX: TStringField;
    tbCustomerTaxRate: TFloatField;
    tbCustomerContact:
```

```
Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBTables, ExtCtrls, QuickRpt, Qrctrls;

type
  TfrmFormLetter = class(TForm)
    QuickRep1: TQuickRep;
    qryEmployee: TQuery;
    DetailBand1: TQRBand;
    qryEmployeeEmpNo: TIntegerField;
    qryEmployeeLastName: TStringField;
    qryEmployeeFirstName: TStringField;
    qryEmployeePhoneExt: TStringField;
    qryEmployeeHireDate: TDateTimeField;
    qryEmployeeSalary: TFloatField;
    QRExprMemo1: TQRExprMemo;
    PageHeaderBand1: TQRBand;
    QRSysData1: TQRSysData;
    <--------> procedure DetailBand1BeforePrint(Sender: TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure QuickRep1BeforePrint(Sender: TCustomQuickRep;
      var PrintReport: Boolean);
  private
    { Private declarations }
    FirstDetail: boolean;
  public
    { Public declarations }
  end;

var
  frmFormLetter: TfrmFormLetter;

implementation

{$R *.dfm}

<--------> procedure TfrmFormLetter.DetailBand1BeforePrint(Sender: TQRCustomBand;
  var PrintBand: Boolean);
begin
  if FirstDetail then
    FirstDetail := False
  else
    QuickRep1.NewPage;
end;

<--------> procedure TfrmFormLetter.QuickRep1BeforePrint(Sender: TCustomQuickRep;
  var PrintReport: Boolean);
begin
  FirstDetail := True;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
5

{ ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
  :: QuickReport 3.0 for Delphi 3.0/4.0/5.0                 ::
  ::                                                        ::
  :: Simple report for grouping data                        ::
  ::                                                        ::
  :: Copyright (c) 1995-1999 QuSoft AS                      ::
  :: All Rights Reserved                                    ::
  ::                                                        ::
  :: web: http://www.qusoft.com  fax: +47 22 41 74 91       ::
  :::::::::::::::::::::::::::::::::::::::::::::::::::::::::: }
unit grouping;

interface

uses
```

```
    { Public declarations }
  end;

var
  frmGrouping: TfrmGrouping;

implementation

{$R *.dfm}

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\

6

{
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
  :: QuickReport 3.0 for Delphi 3.0/4.0/5.0               ::
  ::                                                       ::
  :: Simple report for print the contents of a stringlist  ::
  ::                                                       ::
  :: Copyright (c) 1995-1999 QuSoft AS                    ::
  :: All Rights Reserved                                   ::
  ::                                                       ::
  :: web: http://www.qusoft.com  fax: +47 22 41 74 91      ::
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: }
unit history;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  QuickRpt, Qrctrls, ExtCtrls;

type
  TfrmHistory = class(TForm)
```

```
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBTables, ExtCtrls, QuickRpt, Qrctrls;

type
  TfrmGrouping = class(TForm)
    QuickRep1: TQuickRep;
    MasterQry: TQuery;
    MasterQrycompany: TStringField;
    MasterQryorderno: TFloatField;
    MasterQrycustno: TFloatField;
    MasterQryItemsTotal: TCurrencyField;
    DetailBand1: TQRBand;
    QRDBText2: TQRDBText;
    QRGroup1: TQRGroup;
    QRDBText4: TQRDBText;
    DataSource1: TDataSource;
    DetailQry: TQuery;
    DetailQryOrderNo: TFloatField;
    DetailQryQty: TIntegerField;
    DetailQryItemNo: TFloatField;
    QRSubDetail1: TQRSubDetail;
    QRDBText1: TQRDBText;
    QRDBText5: TQRDBText;
    QRBand1: TQRBand;
    QRExpr1: TQRExpr;
    QRBand2: TQRBand;
    QRExpr2: TQRExpr;
    SummaryBand1: TQRBand;
    QRExpr3: TQRExpr;
    ColumnHeaderBand1: TQRBand;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRLabel7: TQRLabel;
    QRLabel8: TQRLabel;
    PageHeaderBand1: TQRBand;
    QRSysData1: TQRSysData;
    QRSysData2: TQRSysData;
  private
    { Private declarations }
  public
```

```
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  Qrctrls, QuickRpt, Db, DBTables,
ExtCtrls;

type
  TfrmImageRpt = class(TForm)
    QuickRep1: TQuickRep;
    tbBio: TTable;
    DetailBand1: TQRBand;
    tbBioSpeciesNo: TFloatField;
    tbBioCategory: TStringField;
    tbBioCommon_Name:
TStringField;
    tbBioSpeciesName: TStringField;
    tbBioLengthcm: TFloatField;
    tbBioLength_In: TFloatField;
    tbBioNotes: TMemoField;
    tbBioGraphic: TGraphicField;
    QRDBText1: TQRDBText;
    QRDBImage1: TQRDBImage;
    ColumnHeaderBand1: TQRBand;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmImageRpt: TfrmImageRpt;

implementation

{$R *.dfm}

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\
8

{
::::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::
  :: QuickReport 3.0 for Delphi
3.0/4.0/5.0                ::
```

```
    QuickRep1: TQuickRep;
    DetailBand1: TQRBand;
    QRMemo1: TQRMemo;
    PageHeaderBand1: TQRBand;
    QRSysData1: TQRSysData;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmHistory: TfrmHistory;

implementation

{$R *.dfm}

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
7
{
::::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::
  :: QuickReport 3.0 for Delphi
3.0/4.0/5.0                ::
  ::
::
  :: Simple report with graphics
::
  ::
::
  :: Copyright (c) 1995-1999 QuSoft
AS                ::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.com  fax:
+47 22 41 74 91        ::

::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::: }
unit image;

interface

uses
```

```
  ::
::
  :: Master/Detail report with some
extra code              ::
  ::
::
  :: Copyright (c) 1995-1999 QuSoft
AS                  ::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.com  fax:
+47 22 41 74 91        ::

:::::::::::::::::::::::::::::::::::::::::::
:::::::::::::: }
unit mdrpt;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  QuickRpt, Qrctrls, Db, DBTables,
ExtCtrls;

type
  TfrmMasterDetail = class(TForm)
    QuickRep1: TQuickRep;
    DetailBand1: TQRBand;
    dsCustomer: TDataSource;
    qryCustomer: TQuery;
    PageHeaderBand1: TQRBand;
    QRDBText1: TQRDBText;
    QRSysData1: TQRSysData;
    QRSysData2: TQRSysData;
    QRSubDetailOrders:
TQRSubDetail;
    qryOrders: TQuery;
    QRDBText3: TQRDBText;
    QRDBText4: TQRDBText;
    dsOrders: TDataSource;
    qryItems: TQuery;
    QRSubDetailItems:
TQRSubDetail;
    QRDBText5: TQRDBText;
    QRDBText2: TQRDBText;
    QRDBText6: TQRDBText;
    QRDBText7: TQRDBText;
    QRDBText9: TQRDBText;
    QRGroupCust: TQRGroup;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRShapeGray: TQRShape;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRDBOrderNo: TQRDBText;
    QRDBSalesDate: TQRDBText;
    GroupFooterBand1: TQRBand;
    <--------> procedure
QRSubDetailOrdersBeforePrint(Sen
der: TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure
QRSubDetailItemsBeforePrint(Send
er: TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure
QRGroupCustBeforePrint(Sender:
TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure
DetailBand1BeforePrint(Sender:
TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure
QRSubDetailItemsAfterPrint(Sende
r: TQRCustomBand;
      BandPrinted: Boolean);
    <--------> procedure
PageHeaderBand1BeforePrint(Send
er: TQRCustomBand;
      var PrintBand: Boolean);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmMasterDetail:
TfrmMasterDetail;

implementation
```

```
information on the first item subdetail
  // record.  If there are no item records, then we print the subdetail

  // Enable the order fields on the item subdetail band.  After they are
  // printed once, they will be disabled until the next order/item set
  QRDBOrderNo.Enabled := true;
  QRDBSalesDate.Enabled := QRDBOrderNo.Enabled;

  // Only allow this band to print if there are no subdetails
  qryItems.First;
  PrintBand := qryItems.EOF;
end;

<--------> procedure TfrmMasterDetail.QRSubDetailItemsBeforePrint(
  Sender: TQRCustomBand; var PrintBand: Boolean);
begin
  // toggle the item background so that we can have alternating colors
  // like the greenbar paper we all know and love.
  with QRShapeGray.Brush do
    if Color = $00F0F0F0 then
      Color := $00E0E0E0
    else
      Color := $00F0F0F0;
end;

<--------> procedure TfrmMasterDetail.QRSubDetailItemsAfterPrint(
  Sender: TQRCustomBand; BandPrinted: Boolean);
begin
  // After we print it once, we disable the controls
  QRDBOrderNo.Enabled := false;
  QRDBSalesDate.Enabled := QRDBOrderNo.Enabled;
end;

```

```

{$R *.dfm}

<--------> procedure TfrmMasterDetail.DetailBand1BeforePrint(Sender: TQRCustomBand;
  var PrintBand: Boolean);
begin
  // If there are no subdetails, then we print this band.  Otherwise we
  // let the group header print the controls from this band.  Group bands
  // can be reprinted on page breaks, which detail and subdetails can't do.
  qryOrders.First;
  PrintBand := qryOrders.EOF;

  // If it's our turn to print and the group band has our controls, then
  // we take them back.
  if PrintBand and (Sender.ControlCount = 0) then
    with QRGroupCust do
      while ControlCount > 0 do
        Controls[0].Parent := Sender;
end;

<--------> procedure TfrmMasterDetail.QRGroupCustBeforePrint(Sender: TQRCustomBand;
  var PrintBand: Boolean);
begin
  // We grab the detail band fields right from under it.
  if Sender.ControlCount = 0 then
    with DetailBand1 do
      While ControlCount > 0 do
        Controls[0].Parent := Sender;
end;

<--------> procedure TfrmMasterDetail.QRSubDetailOrdersBeforePrint(
  Sender: TQRCustomBand; var PrintBand: Boolean);
begin
  // We are print the order
```

```
ExtCtrls;

type
  TfrmMasterDetail = class(TForm)
    QuickRep1: TQuickRep;
    DetailBand1: TQRBand;
    dsCustomer: TDataSource;
    qryCustomer: TQuery;
    PageHeaderBand1: TQRBand;
    QRDBText1: TQRDBText;
    QRSysData1: TQRSysData;
    QRSysData2: TQRSysData;
    QRSubDetailOrders:
TQRSubDetail;
    qryOrders: TQuery;
    QRDBText3: TQRDBText;
    QRDBText4: TQRDBText;
    dsOrders: TDataSource;
    qryItems: TQuery;
    QRSubDetailItems:
TQRSubDetail;
    QRDBText5: TQRDBText;
    QRDBText2: TQRDBText;
    QRDBText6: TQRDBText;
    QRDBText7: TQRDBText;
    QRDBText9: TQRDBText;
    QRGroupCust: TQRGroup;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRShapeGray: TQRShape;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRDBOrderNo: TQRDBText;
    QRDBSalesDate: TQRDBText;
    GroupFooterBand1: TQRBand;
    <--------> procedure
QRSubDetailOrdersBeforePrint(Sender: TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure
QRSubDetailItemsBeforePrint(Sender: TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure
QRGroupCustBeforePrint(Sender:
TQRCustomBand;
```

```
<--------> procedure
TfrmMasterDetail.PageHeaderBand1BeforePrint(
  Sender: TQRCustomBand; var
PrintBand: Boolean);
begin
  // Re-enable the order fields on the
item subdetail so that the order
  // information will be repeated after
a page break
  QRDBOrderNo.Enabled := true;
  QRDBSalesDate.Enabled :=
QRDBOrderNo.Enabled;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
{
:::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::
  :: QuickReport 3.0 for Delphi
3.0/4.0/5.0                ::
  ::
::
  :: Master/Detail report with some
extra code              ::
  ::
::
  :: Copyright (c) 1995-1999 QuSoft
AS                    ::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.com  fax:
+47 22 41 74 91        ::

:::::::::::::::::::::::::::::::::::::::::::::
::::::::::::: }
unit mdrpt;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  QuickRpt, Qrctrls, Db, DBTables,
```

```
    Controls[0].Parent := Sender;
end;

<--------> procedure
TfrmMasterDetail.QRGroupCustBe
forePrint(Sender:
TQRCustomBand;
  var PrintBand: Boolean);
begin
  // We grab the detail band fields
right from under it.
  if Sender.ControlCount = 0 then
    with DetailBand1 do
      While ControlCount > 0 do
        Controls[0].Parent := Sender;
end;

<--------> procedure
TfrmMasterDetail.QRSubDetailOrd
ersBeforePrint(
  Sender: TQRCustomBand; var
PrintBand: Boolean);
begin
  // We are print the order
information on the first item
subdetail
  // record.  If there are no item
records, then we print the subdetail

  // Enable the order fields on the
item subdetail band.  After they are
  // printed once, they will be
disabled until the next order/item set
  QRDBOrderNo.Enabled := true;
  QRDBSalesDate.Enabled :=
QRDBOrderNo.Enabled;

  // Only allow this band to print if
there are no subdetails
  qryItems.First;
  PrintBand := qryItems.EOF;
end;

<--------> procedure
TfrmMasterDetail.QRSubDetailIte
msBeforePrint(
  Sender: TQRCustomBand; var
PrintBand: Boolean);
begin
```

```
      var PrintBand: Boolean);
    <--------> procedure
DetailBand1BeforePrint(Sender:
TQRCustomBand;
      var PrintBand: Boolean);
    <--------> procedure
QRSubDetailItemsAfterPrint(Sende
r: TQRCustomBand;
      BandPrinted: Boolean);
    <--------> procedure
PageHeaderBand1BeforePrint(Send
er: TQRCustomBand;
      var PrintBand: Boolean);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmMasterDetail:
TfrmMasterDetail;

implementation

{$R *.dfm}

<--------> procedure
TfrmMasterDetail.DetailBand1Befo
rePrint(Sender: TQRCustomBand;
  var PrintBand: Boolean);
begin
  // If there are no subdetails, then we
print this band.  Otherwise we
  // let the group header print the
controls from this band.  Group
bands
  // can be reprinted on page breaks,
which detail and subdetails can't do.
  qryOrders.First;
  PrintBand := qryOrders.EOF;

  // If it's our turn to print and the
group band has our controls, then
  // we take them back.
  if PrintBand and
(Sender.ControlCount = 0) then
    with QRGroupCust do
      while ControlCount > 0 do
```

```
::
  :: Demo report that is populated by
the OnNeedData event   ::
  ::
::
  :: Copyright (c) 1995-1999 QuSoft
AS                    ::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.com  fax:
+47 22 41 74 91        ::

::::::::::::::::::::::::::::::::::::::::::::
::::::::::::: }
unit needdata;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  ExtCtrls, QuickRpt, Qrctrls;

type
  TfrmNeedData = class(TForm)
    QuickRep1: TQuickRep;
    DetailBand1: TQRBand;
    QRLabel1: TQRLabel;
    TitleBand1: TQRBand;
    QRSysData1: TQRSysData;
    <--------> procedure
FormCreate(Sender: TObject);
    <--------> procedure
QuickRep1BeforePrint(Sender:
TCustomQuickRep;
      var PrintReport: Boolean);
    <--------> procedure
QuickRep1NeedData(Sender:
TObject; var MoreData: Boolean);
  private
    { Private declarations }
    SomeList: TStringlist;
    CurrentIndex: integer;
  public
    { Public declarations }
  end;
  // toggle the item background so
that we can have alternating colors
  // like the greenbar paper we all
know and love.
  with QRShapeGray.Brush do
    if Color = $00F0F0F0 then
      Color := $00E0E0E0
    else
      Color := $00F0F0F0;
end;

<--------> procedure
TfrmMasterDetail.QRSubDetailIte
msAfterPrint(
  Sender: TQRCustomBand;
BandPrinted: Boolean);
begin
  // After we print it once, we disable
the controls
  QRDBOrderNo.Enabled := false;
  QRDBSalesDate.Enabled :=
QRDBOrderNo.Enabled;
end;

<--------> procedure
TfrmMasterDetail.PageHeaderBand
1BeforePrint(
  Sender: TQRCustomBand; var
PrintBand: Boolean);
begin
  // Re-enable the order fields on the
item subdetail so that the order
  // information will be repeated after
a page break
  QRDBOrderNo.Enabled := true;
  QRDBSalesDate.Enabled :=
QRDBOrderNo.Enabled;
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
9

{
::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::
  :: QuickReport 3.0 for Delphi
3.0/4.0/5.0             ::
  ::
```

```
  begin
    QRLabel1.Caption :=
SomeList[CurrentIndex];

    // Here's how to set the progress
bar
    QuickRep1.QRPrinter.Progress :=
(Longint(CurrentIndex) * 100) div
SomeList.Count;
  end
  else
    QuickRep1.QRPrinter.Progress :=
100;

  Inc(CurrentIndex);
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
10
{
::::::::::::::::::::::::::::::::::::::::::::
:::::::::::::
  :: QuickReport 3.0 for Delphi
3.0/4.0/5.0/6.0              ::
  ::
::
  :: Example reports project
::
  ::
::
  :: Copyright (c) 1995-1999 QuSoft
AS                    ::
  :: All Rights Reserved
::
  ::
::
  :: web: http://www.qusoft.com  fax:
+47 22 41 74 91        ::

::::::::::::::::::::::::::::::::::::::::::::
::::::::::::: }
unit mdmain;

interface

uses
  Windows, Messages, SysUtils,
```

```

var
  frmNeedData: TfrmNeedData;

implementation

{$R *.dfm}

<--------> procedure
TfrmNeedData.FormCreate(Sender:
TObject);
var
  i: integer;
begin
  SomeList := TStringlist.Create;

  for i := 0 to 500 do
    SomeList.Add('Line ' +
IntToStr(i));
end;

<--------> procedure
TfrmNeedData.QuickRep1BeforePri
nt(Sender: TCustomQuickRep;
  var PrintReport: Boolean);
begin
  // You must reset your data in the
BeforePrint event
  // or when you print from the
preview, the report will
  // start with the last value(s)
  CurrentIndex := 0;
end;

<--------> procedure
TfrmNeedData.QuickRep1NeedData
(Sender: TObject;
  var MoreData: Boolean);
begin
  // If MoreData is true, then
QuickReport will print
  // another detail band.  When you
set it to false,
  // the report is done.

  MoreData := (CurrentIndex <
SomeList.Count);

  if MoreData then
```

```
Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, StdCtrls, quickrpt, Db, DBTables, printers, qrextra, QRExport,
  qrprntr;

const
 Composite_Description = 'The composite control can be used to link several'+
                         ' reports together as a single report.';

type
  TfrmQR3Demo = class(TForm)
    QRTextFilter1: TQRTextFilter;
    QRCSVFilter1: TQRCSVFilter;
    QRHTMLFilter1: TQRHTMLFilter;
    QRCompositeReport1: TQRCompositeReport;
    Label1: TLabel;
    VersionLbl: TLabel;
    Label3: TLabel;
    GroupBox1: TGroupBox;
    rbCreateList: TRadioButton;
    btnPreview: TButton;
    btnPrint: TButton;
    rbMasterDetail: TRadioButton;
    Description: TMemo;
    OpenDialog1: TOpenDialog;
    cbPreview: TComboBox;
    Label4: TLabel;
    rbExprMemo: TRadioButton;
    rbImage: TRadioButton;
    rbBasicMD: TRadioButton;
    btnFilters: TButton;
    rbComposite: TRadioButton;
    Label5: TLabel;
    rbNeedData: TRadioButton;
    rbFormLetter: TRadioButton;
    btnExport: TButton;
    SaveDialog1: TSaveDialog;
    rbAbout: TRadioButton;
    rbGrouping: TRadioButton;
    Image2: TImage;
    <--------> procedure
CreateList(Sender: TObject);
    <--------> procedure
QRCompositeReport1AddReports(Sender: TObject);
    <--------> procedure
btnCRClick(Sender: TObject);
    <--------> procedure
rbCreateListClick(Sender: TObject);
    <--------> procedure
rbMasterDetailClick(Sender: TObject);
    <--------> procedure
FormCreate(Sender: TObject);
    <--------> procedure
FormActivate(Sender: TObject);
    <--------> procedure
btnPreviewClick(Sender: TObject);
    <--------> procedure
rbExprMemoClick(Sender: TObject);
    <--------> procedure
rbImageClick(Sender: TObject);
    <--------> procedure
rbBasicMDClick(Sender: TObject);
    <--------> procedure
btnFiltersClick(Sender: TObject);
    <--------> procedure
rbCompositeClick(Sender: TObject);
    <--------> procedure
rbNeedDataClick(Sender: TObject);
    <--------> procedure
rbFormLetterClick(Sender: TObject);
    <--------> procedure
btnExportClick(Sender: TObject);
    <--------> procedure
rbAboutClick(Sender: TObject);
    <--------> procedure
rbGroupingClick(Sender: TObject);
  private
    { Private declarations }
    FReport : TCustomQuickRep;
    CreateListReport : TQuickRep;
    <--------> procedure
SetReport(Value : TCustomQuickRep);
  public
```

```
from the table }
  SomeFields := TStringList.Create;
  with MyTable do
  begin
    DatabaseName := 'DBDEMOS';
    TableName := 'COUNTRY.DB';
    ReadOnly := True;
    Active := True;

    // For this example, we will pull the field names from the table
    // If you wanted to only use some of the fields, you would edit
    // this list.
    for nIdx := 0 to FieldCount - 1 do
      SomeFields.Add(Fields[nIdx].FieldName);
  end;

  // If you didn't create the report, you must set the report object to nil
  // before calling QRCreateList

  areport := nil;

  { Build the report }

  // If you change the displaywidth, it will be reflecte in the created
  // report

  with MyTable.Fields[1] do
    DisplayWidth := DisplayWidth div 2;

  // create the report
  QRCreateList(aReport, nil, MyTable, 'Country Listing', SomeFields);

  // Make the column header's font use bold attribute
  aReport.Bands.ColumnHeaderBand.Font.Style := [fsBold];

  // Now adjust the spacing of the fields.  There isn't any reason to
    { Public declarations }
    property Report : TCustomQuickRep read FReport write SetReport;
  end;

var
  frmQR3Demo: TfrmQR3Demo;

implementation

uses mdrpt, exprmemo, image, basicmd, needdata, frmltr,
  history, grouping;

{$R *.dfm}

<--------> procedure TfrmQR3Demo.SetReport(Value : TCustomQuickRep);
begin
  FReport := Value;
  if Value <> nil then
    if Value = TCustomQuickRep(QRCompositeReport1) then
      Description.Lines.Text := Composite_Description
    else
      Description.Lines.Assign(Report.Description);
end;

<--------> procedure TfrmQR3Demo.CreateList(Sender: TObject);
var
  aReport : TCustomQuickRep;
  SomeFields: TStringList;
  MyTable: TTable;
  nIdx: integer;
begin
  { Create a table on the fly, this example uses a table from the demo database }
  MyTable := TTable.Create(self);

  { create the list of fields to output
```

```
end;

<--------> procedure TfrmQR3Demo.QRCompositeReport1AddReports(Sender: TObject);
begin
  // The OnAddReports event is called by the CompositeReport
  // to add the reports to list of reports
  with QRCompositeReport1.Reports do
  begin
    Add(frmMasterDetail.QuickRep1);
    Add(frmBasicMD.QuickRep1);
    Add(frmImageRpt.QuickRep1);
  end;
end;

<--------> procedure TfrmQR3Demo.btnCRClick(Sender : TObject);
begin
  QRCompositeReport1.Preview;
end;

<--------> procedure TfrmQR3Demo.rbCreateListClick(Sender: TObject);
begin
  Report := CreateListReport;
end;

<--------> procedure TfrmQR3Demo.rbMasterDetailClick(Sender: TObject);
begin
  Report := frmMasterDetail.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.FormCreate(Sender : TObject);
var i: integer;
begin
  // Get the current QuickReport version number from
```

```
  // do this, this is just to show how to access the controls on the
  // report.

  for nIdx := 0 to aReport.Bands.ColumnHeaderBand.ControlCount -1 do
    if aReport.Bands.ColumnHeaderBand.Controls[nIdx] is TQRPrintable then
      with TQRPrintable(aReport.Bands.ColumnHeaderBand.Controls[nIdx]) do
        Left := Left - (5 * nIdx);

  for nIdx := 0 to aReport.Bands.DetailBand.ControlCount -1 do
    if aReport.Bands.DetailBand.Controls[nIdx] is TQRPrintable then
      with TQRPrintable(aReport.Bands.DetailBand.Controls[nIdx]) do
        Left := Left - (5 * nIdx);


  { You can change the report objects before calling the report }
//  areport.page.orientation := poLandscape;
  {preview or print the report}

  if sender = btnPreview then
  begin
    case cbPreview.ItemIndex of
      0: areport.preview;
      1: areport.previewModal;
      2: areport.previewModeless;
    end;
  end
  else if sender = btnPrint then
    areport.print;

  { all done, free the objects }
  aReport.Free;
  MyTable.Free;
  SomeFields.Free;
```

```
    QRCompositeReport1.print;
  end
  else
  begin
    if sender = btnPreview then
    begin
      case cbPreview.ItemIndex of
        0: report.preview;
        1: report.previewModal;
        2: report.previewModeless;
      end;
    end
    else if sender = btnPrint then
      report.print;
  end;
 end;
end;

<--------> procedure
TfrmQR3Demo.rbExprMemoClick(
Sender: TObject);
begin
  Report :=
frmExprmemo.QuickRep1;
end;

<--------> procedure
TfrmQR3Demo.rbImageClick(Send
er: TObject);
begin
  Report :=
frmImageRpt.QuickRep1;
end;

<--------> procedure
TfrmQR3Demo.rbBasicMDClick(Se
nder: TObject);
begin
  Report :=
frmBasicMD.QuickRep1;
end;

<--------> procedure
TfrmQR3Demo.btnFiltersClick(Sen
der: TObject);
begin
  MessageDlg('When an export filter
control is dropped on a form, it''s
added to all of the
```

```
  // the qrprntr unit
  i := pos(' ', cQRName);
  if i >=0 then
    VersionLbl.Caption := 'Version' +
copy(cQRName, i, 99);
  // Create the report object used by
the QRCreateList() function
  CreateListReport :=
TQuickRep.Create(self);
  CreateListReport.Description.Text
:= 'Example of how to call the
QRCreateList function';
  cbPreview.ItemIndex := 0;
end;

<--------> procedure
TfrmQR3Demo.FormActivate(Send
er: TObject);
begin
  if Description.Lines.Count = 0 then
  begin
    rbCreateListClick(Self);
    rbCreateList.Checked := True;
  end;
end;

<--------> procedure
TfrmQR3Demo.btnPreviewClick(Se
nder: TObject);
begin
  // This code is more complicated
than what you would
  // typically need to run a report.
Most of this code
  // is to handle the selection of the
various types of
  // reports.

  if report <> nil then
  begin
    if report = CreateListReport then
      CreateList(Sender)
    else if report =
TCustomQuickRep(QRCompositeR
eport1) then
    begin
      if sender = btnPreview then
        QRCompositeReport1.preview
      else
```

```
ASCII: TQRAsciiExportFilter
CSV: TQRCommaSeparatedFilter

In Professional Version:
RTF: TQRRTFExportFilter
WMF: TQRWMFExportFilter
Excel: TQRXLSFilter
}
    end;
  end;
  btnExport.Enabled := True;
end;

<--------> procedure TfrmQR3Demo.rbAboutClick(Sender: TObject);
begin
  Report := frmHistory.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.rbGroupingClick(Sender: TObject);
begin
  Report := frmGrouping.QuickRep1;
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

<------> Objects83

الملف النصي لبرنامج

(Qr3)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<------> Object frmQR3Demo: TfrmQR3Demo
  Left = 182
  Top = 108
  Width = 603
  Height = 340
  Caption = 'Quick Report 3
```

```
previews',mtInformation, [mbok], 0);
end;

<--------> procedure TfrmQR3Demo.rbCompositeClick(Sender: TObject);
begin
  Report := TCustomQuickRep(QRCompositeReport1);
end;

<--------> procedure TfrmQR3Demo.rbNeedDataClick(Sender: TObject);
begin
  Report := frmNeedData.QuickRep1;
end;

<--------> procedure TfrmQR3Demo.rbFormLetterClick(Sender: TObject);
begin
  Report := frmFormLetter.QuickRep1;
end;

// The following code show how to explicitly call an export
// filter without going through the preview
<--------> procedure TfrmQR3Demo.btnExportClick(Sender: TObject);
begin
  btnExport.Enabled := False;
  with SaveDialog1 do
  begin
    if Execute then
    begin

frmFormLetter.QuickRep1.ExportToFilter(TQRCommaSeparatedFilter.Create(FileName));
{
Other filters:
HTML: TQRHTMLDocumentFilter
```

Standard Demo for Delphi 6'
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
Scaled = False
OnActivate = FormActivate
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<------> Object Label1: TLabel
Left = 56
Top = 4
Width = 174
Height = 34
Caption = 'QuickReport'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -29
Font.Name = 'Arial'
Font.Style = [fsBold]
ParentFont = False
Transparent = True
end
<------> Object VersionLbl: TLabel
Left = 236
Top = 18
Width = 70
Height = 16
Caption = 'Version 3.0'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'Arial'
Font.Style = [fsBold]
ParentFont = False
Transparent = True
end
<------> Object Label3: TLabel
Left = 56
Top = 36

Width = 279
Height = 13
Caption = 'Copyright 1995 - 1999 QuSoft AS http://www.qusoft.com'
Transparent = True
end
<------> Object Label4: TLabel
Left = 24
Top = 272
Width = 61
Height = 13
Alignment = taRightJustify
Caption = 'Preview type'
end
<------> Object Label5: TLabel
Left = 56
Top = 50
Width = 158
Height = 13
Caption = 'Demo version 3.1.0 July 13, 1999'
Transparent = True
end
<------> Object Image2: TImage
Left = 6
Top = 6
Width = 47
Height = 47
Picture.Data = {
Transparent = True
end
<------> Object GroupBox1: TGroupBox
Left = 7
Top = 64
Width = 586
Height = 193
Caption = 'Select a Report'
TabOrder = 0
<------> Object rbCreateList: TRadioButton
Left = 4
Top = 32
Width = 179
Height = 17
Caption = 'QRCreatelist() example'

```
    Top = 49
    Width = 197
    Height = 17
    Caption = 'Basic Master/Detail example'
    TabOrder = 3
    OnClick = rbBasicMDClick
  end
  <------> Object rbComposite: TRadioButton
    Left = 4
    Top = 135
    Width = 179
    Height = 17
    Caption = 'Composite Report'
    TabOrder = 8
    OnClick = rbCompositeClick
  end
  <------> Object rbNeedData: TRadioButton
    Left = 4
    Top = 152
    Width = 179
    Height = 17
    Caption = 'OnNeedData example'
    TabOrder = 9
    OnClick = rbNeedDataClick
  end
  <------> Object rbFormLetter: TRadioButton
    Left = 4
    Top = 118
    Width = 179
    Height = 17
    Caption = 'Form Letter example'
    TabOrder = 7
    OnClick = rbFormLetterClick
  end
  <------> Object rbAbout: TRadioButton
    Left = 4
    Top = 15
    Width = 179
    Height = 17
    Caption = 'QR3DEMO History'
    TabOrder = 1
    OnClick = rbAboutClick
```

```
    TabOrder = 2
    OnClick = rbCreateListClick
  end
  <------> Object rbMasterDetail: TRadioButton
    Left = 4
    Top = 66
    Width = 179
    Height = 17
    Caption = 'Master/Detail example'
    TabOrder = 4
    OnClick = rbMasterDetailClick
  end
  <------> Object Description: TMemo
    Left = 207
    Top = 16
    Width = 370
    Height = 153
    ReadOnly = True
    ScrollBars = ssVertical
    TabOrder = 0
  end
  <------> Object rbExprMemo: TRadioButton
    Left = 4
    Top = 101
    Width = 179
    Height = 17
    Caption = 'ExprMemo example'
    TabOrder = 6
    OnClick = rbExprMemoClick
  end
  <------> Object rbImage: TRadioButton
    Left = 4
    Top = 83
    Width = 179
    Height = 17
    Caption = 'List report with bitmaps'
    TabOrder = 5
    OnClick = rbImageClick
  end
  <------> Object rbBasicMD: TRadioButton
    Left = 4
```

```
    Left = 392
    Top = 5
    Width = 113
    Height = 25
    Caption = 'About the filters..'
    TabOrder = 4
    OnClick = btnFiltersClick
  end
  <------> Object btnExport: TButton
    Left = 488
    Top = 268
    Width = 75
    Height = 25
    Caption = 'CSV Export'
    TabOrder = 5
    OnClick = btnExportClick
  end
  <------> Object OpenDialog1:
TOpenDialog
    DefaultExt = 'TXT'
    Filter = 'Text files (*.txt)|*.TXT'
    Left = 240
    Top = 155
  end
  <------> Object SaveDialog1:
TSaveDialog
    DefaultExt = 'RTF'
    Filter = 'Richtext files (*.rtf)|*.rtf'
    Title = 'Export RTF'
    Left = 240
    Top = 91
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\
2
<------> Object frmNeedData:
TfrmNeedData
  Left = 192
  Top = 107
  Width = 696
  Height = 480
  Caption = 'frmNeedData'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
```

```
    end
    <------> Object rbGrouping:
TRadioButton
      Left = 4
      Top = 169
      Width = 179
      Height = 17
      Caption = 'Grouping and
subtotaling'
      TabOrder = 10
      OnClick = rbGroupingClick
    end
  end
  <------> Object btnPreview:
TButton
    Left = 360
    Top = 268
    Width = 59
    Height = 25
    Caption = 'Preview'
    TabOrder = 1
    OnClick = btnPreviewClick
  end
  <------> Object btnPrint: TButton
    Left = 424
    Top = 268
    Width = 57
    Height = 25
    Caption = 'Print'
    TabOrder = 2
    OnClick = btnPreviewClick
  end
  <------> Object cbPreview:
TComboBox
    Left = 89
    Top = 268
    Width = 265
    Height = 21
    Style = csDropDownList
    ItemHeight = 13
    TabOrder = 3
    Items.Strings = (
      'Standard modeless preview
(non-threaded)'
      'Modal (runs in a thread)'
      'Modeless  (runs in a thread)')
  end
  <------> Object btnFilters: TButton
```

```
   SQL.Strings = (
     'select * from orders'
     'where CustNo = :CustNo'
     'order by CustNo, OrderNo')
   Left = 48
   ParamData = <
     item
       DataType = ftFloat
       Name = 'CustNo'
       ParamType = ptUnknown
       Size = 8
     end>
 end
 <------> Object dsOrders:
TDataSource
   DataSet = qryOrders
   Top = 27
 end
 <------> Object qryItems: TQuery
   Active = True
   DatabaseName = 'DBDEMOS'
   DataSource = dsOrders
   SQL.Strings = (
     'SELECT * '
     'FROM items'
     'WHERE OrderNo = :OrderNo'
     'ORDER BY OrderNo, ItemNo')
   Left = 72
   ParamData = <
     item
       DataType = ftFloat
       Name = 'OrderNo'
       ParamType = ptUnknown
       Size = 8
     end>
 end
End

\\\\\\\\\\\\\\\\\\\\\\

4
<------> Object frmImageRpt:
TfrmImageRpt
 Left = 192
 Top = 107
 Width = 696
 Height = 480
 Caption = 'frmImageRpt'
```

```
 Font.Style = []
 OldCreateOrder = False
 Scaled = False
 OnCreate = FormCreate
 PixelsPerInch = 96
 TextHeight = 13
End
\\\\\\\\\\\\\\\\\\\\\\


3

<------> Object frmMasterDetail:
TfrmMasterDetail
 Left = 153
 Top = 119
 Width = 799
 Height = 480
 Caption = 'frmMasterDetail'
 Color = clBtnFace
 Font.Charset =
DEFAULT_CHARSET
 Font.Color = clWindowText
 Font.Height = -11
 Font.Name = 'MS Sans Serif'
 Font.Style = []
 OldCreateOrder = True
 Scaled = False
 PixelsPerInch = 96
 TextHeight = 13
 <------> Object dsCustomer:
TDataSource
   DataSet = qryCustomer
 end
 <------> Object qryCustomer:
TQuery
   Active = True
   DatabaseName = 'DBDEMOS'
   SQL.Strings = (
     'select * from customer'
     'Where CustNo <= 1354'
     'order by CustNo')
   Left = 24
 end
 <------> Object qryOrders: TQuery
   Active = True
   DatabaseName = 'DBDEMOS'
   DataSource = dsCustomer
```

```
    Size = 50
   end
   <------> Object tbBioGraphic:
TGraphicField
    FieldName = 'Graphic'
    BlobType = ftGraphic
   end
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
5

<------> Object frmHistory:
TfrmHistory
  Left = 192
  Top = 107
  Width = 696
  Height = 480
  Caption = 'frmHistory'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  Scaled = False
  PixelsPerInch = 96
  TextHeight = 13
End

\\\\\\\\\\\\\\\\\\\\\\\\\\

6
<------> Object frmGrouping:
TfrmGrouping
  Left = 192
  Top = 107
  Width = 774
  Height = 480
  Caption = 'frmGrouping'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
```

```
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  Scaled = False
  PixelsPerInch = 96
  TextHeight = 13
  <------> Object tbBio: TTable
   Active = True
   DatabaseName = 'DBDEMOS'
   TableName = 'BIOLIFE.DB'
   Left = 8
   Top = 16
   <------> Object tbBioSpeciesNo:
TFloatField
    FieldName = 'Species No'
   end
   <------> Object tbBioCategory:
TStringField
    FieldName = 'Category'
    Size = 15
   end
   <------> Object
tbBioCommon_Name: TStringField
    FieldName = 'Common_Name'
    Size = 30
   end
   <------> Object tbBioSpeciesName:
TStringField
    FieldName = 'Species Name'
    Size = 40
   end
   <------> Object tbBioLengthcm:
TFloatField
    FieldName = 'Length (cm)'
   end
   <------> Object tbBioLength_In:
TFloatField
    FieldName = 'Length_In'
   end
   <------> Object tbBioNotes:
TMemoField
    FieldName = 'Notes'
    BlobType = ftMemo
```

```
 <------> Object DetailQry: TQuery
   Active = True
   DatabaseName = 'DBDEMOS'
   DataSource = DataSource1
   SQL.Strings = (
     'SELECT OrderNo, Qty, ItemNo'
     'FROM ITEMS I'
     'WHERE (i.OrderNo = :OrderNo)'
     'ORDER BY OrderNo, ItemNo')
   Left = 8
   Top = 64
   ParamData = <
     item
       DataType = ftFloat
       Name = 'orderno'
       ParamType = ptUnknown
       Size = 8
     end>
   <------> Object DetailQryOrderNo: TFloatField
     FieldName = 'OrderNo'
     Origin = '"ITEMS.DB".OrderNo'
   end
   <------> Object DetailQryQty: TIntegerField
     FieldName = 'Qty'
     Origin = '"ITEMS.DB".Qty'
   end
   <------> Object DetailQryItemNo: TFloatField
     FieldName = 'ItemNo'
     Origin = '"ITEMS.DB".ItemNo'
   end
 end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
7

<------> Object frmFormLetter: TfrmFormLetter
 Left = 200
 Top = 108
 Width = 544
 Height = 375
 Font.Style = []
 OldCreateOrder = False
 Scaled = False
 PixelsPerInch = 96
 TextHeight = 13
 <------> Object MasterQry: TQuery
   Active = True
   DatabaseName = 'DBDEMOS'
   SQL.Strings = (
     'select c.company, o.orderno, o.custno, o.ItemsTotal'
     'from orders o join customer c on o.custno=c.custno'
     'where c.company LIKE "A%"'
     'order by c.company, o.orderno')
   Left = 8
   <------> Object MasterQrycompany: TStringField
     FieldName = 'company'
     Origin = '"CUSTOMER.DB".Company'
     Size = 30
   end
   <------> Object MasterQryorderno: TFloatField
     FieldName = 'orderno'
     Origin = '"ORDERS.DB".OrderNo'
   end
   <------> Object MasterQrycustno: TFloatField
     FieldName = 'custno'
     Origin = '"ORDERS.DB".CustNo'
   end
   <------> Object MasterQryItemsTotal: TCurrencyField
     FieldName = 'ItemsTotal'
     Origin = '"ORDERS.DB".ItemsTotal'
   end
 end
 <------> Object DataSource1: TDataSource
   DataSet = MasterQry
   Left = 8
   Top = 32
 end
```

end
<------> Object qryEmployeePhoneExt: TStringField
FieldName = 'PhoneExt'
Origin = '"Employee.DB".PhoneExt'
Size = 4
end
<------> Object qryEmployeeHireDate: TDateTimeField
FieldName = 'HireDate'
Origin = '"Employee.DB".HireDate'
end
<------> Object qryEmployeeSalary: TFloatField
CustomConstraint = 'X > 4499'
ConstraintErrorMessage = 'Minimum Salary is $4,500.00'
FieldName = 'Salary'
Origin = '"Employee.DB".Salary'
currency = True
end
end
End
\\
8
<------> Object frmExprmemo: TfrmExprmemo
Left = 192
Top = 107
Width = 696
Height = 480
Caption = 'frmExprmemo'
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = False
Scaled = False
PixelsPerInch = 96
TextHeight = 13

Caption = 'frmFormLetter'
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Scaled = False
PixelsPerInch = 96
TextHeight = 13
<------> Object qryEmployee: TQuery
Active = True
DatabaseName = 'DBDEMOS'
SQL.Strings = (
'SELECT *'
'FROM Employee'
'WHERE HireDate < "01/01/1990"')
Left = 8
Top = 8
<------> Object qryEmployeeEmpNo: TIntegerField
CustomConstraint = 'Value > 0'
ConstraintErrorMessage = 'EmpNo cannot be 0 or negative'
FieldName = 'EmpNo'
Origin = '"Employee.DB".EmpNo'
DisplayFormat = 'Emp"#" 0000'
MaxValue = 9999
MinValue = 1
end
<------> Object qryEmployeeLastName: TStringField
FieldName = 'LastName'
Origin = '"Employee.DB".LastName'
end
<------> Object qryEmployeeFirstName: TStringField
FieldName = 'FirstName'
Origin = '"Employee.DB".FirstName'
Size = 15

```
    FieldName = 'Country'
  end
  <------> Object tbCustomerPhone: TStringField
    FieldName = 'Phone'
    Size = 15
  end
  <------> Object tbCustomerFAX: TStringField
    FieldName = 'FAX'
    Size = 15
  end
  <------> Object tbCustomerTaxRate: TFloatField
    FieldName = 'TaxRate'
    DisplayFormat = '0.00%'
  end
  <------> Object tbCustomerContact: TStringField
    FieldName = 'Contact'
  end
  <------> Object tbCustomerLastInvoiceDate: TDateTimeField
    FieldName = 'LastInvoiceDate'
  end
 end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

9
<------> Object frmBasicMD: TfrmBasicMD
 Left = 192
 Top = 107
 Width = 696
 Height = 480
 Caption = 'frmBasicMD'
 Color = clBtnFace
 Font.Charset = DEFAULT_CHARSET
 Font.Color = clWindowText
 Font.Height = -11
 Font.Name = 'MS Sans Serif'
 Font.Style = []
 OldCreateOrder = False
 Scaled = False
 <------> Object tbCustomer: TTable
   Active = True
   DatabaseName = 'DBDEMOS'
   IndexFieldNames = 'CustNo'
   TableName = 'CUSTOMER.DB'
   Left = 32
   Top = 24
   <------> Object tbCustomerCustNo: TFloatField
     Alignment = taLeftJustify
     FieldName = 'CustNo'
     DisplayFormat = 'CN 0000'
     MaxValue = 9999.000000000000000000
     MinValue = 1000.000000000000000000
   end
   <------> Object tbCustomerCompany: TStringField
     FieldName = 'Company'
     Size = 30
   end
   <------> Object tbCustomerAddr1: TStringField
     FieldName = 'Addr1'
     Size = 30
   end
   <------> Object tbCustomerAddr2: TStringField
     FieldName = 'Addr2'
     Size = 30
   end
   <------> Object tbCustomerCity: TStringField
     FieldName = 'City'
     Size = 15
   end
   <------> Object tbCustomerState: TStringField
     FieldName = 'State'
   end
   <------> Object tbCustomerZip: TStringField
     FieldName = 'Zip'
     Size = 10
   end
   <------> Object tbCustomerCountry: TStringField
```

```
  PixelsPerInch = 96
  TextHeight = 13
  <------> Object dsCustomer:
TDataSource
    DataSet = tbCustomer
    Left = 24
  end
  <------> Object tbCustomer:
TTable
    Active = True
    DatabaseName = 'DBDEMOS'
    IndexFieldNames = 'Company'
    TableName = 'CUSTOMER.DB'
    Left = 24
    Top = 32
  end
  <------> Object qryOrders: TQuery
    Active = True
    DatabaseName = 'DBDEMOS'
    DataSource = dsCustomer
    SQL.Strings = (
      'select * from orders '
      'WHERE (CustNo = :CustNo)'
      'order by custno, Terms')
    Left = 24
    Top = 56
    ParamData = <
      item
        DataType = ftFloat
        Name = 'CustNo'
        ParamType = ptUnknown
        Size = 8
      end>
  end
End

\\\\\\\\\\\\\\\\\\
10
```

القسم الثالث

برامج منوعة في دلفي

Procedures 4

الملف التنفيذي لبرنامج

(BS Morph Button)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  BSMorphButton, StdCtrls;

type
  TForm1 = class(TForm)
    BSMorphButton1:
TBSMorphButton;
    Label1: TLabel;
    BSMorphButton2:
TBSMorphButton;
    Label2: TLabel;
    BSMorphButton3:
TBSMorphButton;
    Label3: TLabel;
    Label4: TLabel;
    BSMorphButton4:
TBSMorphButton;
    BSMorphButton5:
TBSMorphButton;
    <-------> procedure
BSMorphButton1Click(Sender:
TObject);
    <-------> procedure
BSMorphButton5Click(Sender:
TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

<-------> procedure
TForm1.BSMorphButton1Click(Sender: TObject);
begin
 ShowMessage ('         '
            +(Sender as
TBSMorphButton).Hint
            +'         ');
end;

<-------> procedure
TForm1.BSMorphButton5Click(Sender: TObject);
begin
 Close;
end;

end.
```

<-------> Objects5

الملف النصي لبرنامج

(BS Morph Button)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<-------> Object Form1: TForm1
  Left = 158
  Top = 159
  BorderStyle = bsToolWindow
  Caption = 'BSMorphButton
DEMO'
  ClientHeight = 129
  ClientWidth = 520
```

```
  Color = clBlack
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  Position = poScreenCenter
  PixelsPerInch = 96
  TextHeight = 13
  <-------> Object Label1: TLabel
    Left = 8
    Top = 9
    Width = 120
    Height = 13
    Alignment = taCenter
    AutoSize = False
    Caption = 'S H A D O W'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clWhite
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    Transparent = True
  end
  <-------> Object Label2: TLabel
    Left = 136
    Top = 9
    Width = 120
    Height = 13
    Alignment = taCenter
    AutoSize = False
    Caption = 'C O L O R S'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clWhite
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    Transparent = True
  end
  <-------> Object Label3: TLabel
    Left = 264
    Top = 9
```

```
    Width = 120
    Height = 13
    Alignment = taCenter
    AutoSize = False
    Caption = 'L I G H T S'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clWhite
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    Transparent = True
  end
  <-------> Object Label4: TLabel
    Left = 392
    Top = 9
    Width = 120
    Height = 13
    Alignment = taCenter
    AutoSize = False
    Caption = 'M O V E'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clWhite
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    Transparent = True
  end
End
```

الملف التنفيذي لبرنامج

(BatchMv)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
Procedures 28

unit Batmove;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls, ExtCtrls, Db, DBTables,
ComCtrls;

type
  TForm1 = class(TForm)
    BatchMove1: TBatchMove;
    tblSource: TTable;
    tblDest: TTable;
    Bevel1: TBevel ;
    Label6: TLabel;
    cmbxDestAlias: TComboBox;
    Label7: TLabel;
    cmbxDestTable: TComboBox;
    Label8: TLabel;
    cmbxDestIndex: TComboBox;
    Bevel2: TBevel;
    cmbxSourceAlias: TComboBox;
    cmbxSourceTable: TComboBox;
    cmbxSourceIndex: TComboBox;
    Label5: TLabel;
    Label4: TLabel;
    Label3: TLabel;
    cmbxMode: TComboBox;
    Button1: TButton;
    Bevel3: TBevel;
    chkbxAbortKey: TCheckBox;
    chkbxAbortProblem: TCheckBox;
    chkbxTrans: TCheckBox;
    Label1: TLabel;
    Label2: TLabel;
    Label10: TLabel;
    Label9: TLabel;
    Label12: TLabel;
    edtChangedTable: TEdit;
    Label13: TLabel;
    edtKeyVioTbl: TEdit;
    Label14: TLabel;
    edtProbTbl: TEdit;
    Label11: TLabel;
    edtRecCount: TEdit;
    <------->procedure
FormCreate(Sender: TObject);
    <------->procedure
cmbxSourceAliasChange(Sender:
TObject);
    <------->procedure
cmbxDestAliasChange(Sender:
TObject);
    <------->procedure
cmbxSourceTableChange(Sender:
TObject);
    <------->procedure
cmbxDestTableChange(Sender:
TObject);
    <------->procedure
cmbxSourceIndexChange(Sender:
TObject);
    <------->procedure
cmbxDestIndexChange(Sender:
TObject);
    <------->procedure
Button1Click(Sender: TObject);
    <------->procedure
chkbxAbortKeyClick(Sender:
TObject);
    <------->procedure
chkbxAbortProblemClick(Sender:
TObject);
```

```
Session.GetTableNames(tblSource.DatabaseName,'',true,false,cmbxSourceTable.Items);
  end
  else
  begin
   tblSource.DatabaseName := '';
   cmbxSourceTable.Items.Clear;
  end;
end;

<------->procedure TForm1.cmbxDestAliasChange(Sender: TObject);
begin
  if cmbxDestAlias.ItemIndex <> -1 then
  begin
    tblDest.DatabaseName := cmbxDestAlias.Items[cmbxDestAlias.ItemIndex];
Session.GetTableNames(tblDest.DatabaseName,'',true,false,cmbxDestTable.Items);
  end
  else
  begin
   tblDest.DatabaseName := '';
   cmbxDestTable.Items.Clear;
  end;
end;

<------->procedure TForm1.cmbxSourceTableChange(Sender: TObject);
begin
  if cmbxSourceTable.ItemIndex <> -1 then
  begin
    tblSource.TableName := cmbxSourceTable.Items[cmbxSourceTable.ItemIndex];
tblSource.GetIndexNames(cmbxSourceIndex.Items);
  end
  else
```

```
   <------->procedure chkbxTransClick(Sender: TObject);
   <------->procedure cmbxModeChange(Sender: TObject);
   <------->procedure edtRecCountKeyPress(Sender: TObject; var Key: Char);
   <------->procedure FormShow(Sender: TObject);
  private
    { Private declarations }
    function IsStringsEqual(const s1,s2 : string): boolean;
//    simple utility function
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

<------->procedure TForm1.FormCreate(Sender: TObject); // Get alias names
begin
  if Session.Active = false then
    Session.Open;
Session.GetAliasNames(CmbxSourceAlias.Items);
  cmbxDestAlias.Items := cmbxSourceAlias.Items;
end;

<------->procedure TForm1.cmbxSourceAliasChange(Sender: TObject);
begin
  if cmbxSourceAlias.ItemIndex <> -1 then
  begin
    tblSource.DatabaseName := cmbxSourceAlias.Items[cmbxSourceAlias.ItemIndex];
```

```
  if cmbxDestIndex.ItemIndex <> -1
then
  begin
    tblDest.IndexName :=
cmbxDestIndex.Items[cmbxDestInde
x.ItemIndex];
  end
  else
  begin
    tblDest.IndexName := '';
  end;

end;

<------->procedure
TForm1.Button1Click(Sender:
TObject);
begin
  if tblDest.TableName = '' then
    tblDest.TableName :=
cmbxDestTable.Text;
  if ((tblSource.DatabaseName <> '')
and // test for enough input
     (tblSource.TableName <> '') and
     (tblDest.DatabaseName <> '') and
     (tblDest.TableName <> '') and

(cmbxMode.items[cmbxMode.ItemI
ndex] <> '')) then
  begin
    BatchMove1.ChangedTableName
:= edtChangedTable.Text;  // more
batchmove setup
    BatchMove1.KeyViolTableName
:= edtKeyVioTbl.Text;
    BatchMove1.ProblemTableName
:= edtProbTbl.Text;
    BatchMove1.RecordCount :=
StrToInt(edtRecCount.Text);
  end
  else
  begin
    MessageDlg('Incomplete
input.',mtError,[mbOK],0);
    exit;
  end;
  BatchMove1.Execute;  // run the
batchmove
  MessageDlg('BatchMove complete.
```

```
  begin
    tblSource.TableName := '';
    cmbxSourceIndex.Items.Clear;
  end;

end;

<------->procedure
TForm1.cmbxDestTableChange(Sen
der: TObject);
begin
  if cmbxDestTable.ItemIndex <> -1
then
  begin
     tblDest.TableName :=
cmbxDestTable.Items[cmbxDestTab
le.ItemIndex];

tblDest.GetIndexNames(cmbxDestIn
dex.Items);
  end
  else
  begin
    tblDest.TableName := '';
    cmbxDestIndex.Items.Clear;
  end;
end;

<------->procedure
TForm1.cmbxSourceIndexChange(S
ender: TObject);
begin
  if cmbxSourceIndex.ItemIndex <> -
1 then
  begin
    tblSource.IndexName :=
cmbxSourceIndex.Items[cmbxSourc
eIndex.ItemIndex];
  end
  else
  begin
     tblSource.IndexName := '';
  end;
end;

<------->procedure
TForm1.cmbxDestIndexChange(Sen
der: TObject);
begin
```

```
    else if IsStringsEqual(cmbxMode.Items[cmbxMode.ItemIndex],'Copy') then
      BatchMove1.Mode := batCopy
    else if IsStringsEqual(cmbxMode.Items[cmbxMode.ItemIndex],'Append Update') then
      BatchMove1.Mode := batAppendUpdate
    else if IsStringsEqual(cmbxMode.Items[cmbxMode.ItemIndex],'Delete') then
      BatchMove1.Mode := batDelete
    else if IsStringsEqual(cmbxMode.Items[cmbxMode.ItemIndex],'Update') then
      BatchMove1.Mode := batUpdate
    else
      MessageDlg('Batch mode not found',mtError,[mbOK],0);
  end;
end;


// only allow numbers to be typed in
<------->procedure TForm1.edtRecCountKeyPress(Sender: TObject; var Key: Char);
begin
  if ((key in ['0'..'9'] = false) and (word(key) <> VK_BACK)) then
    key := #0;
end;

<------->procedure TForm1.FormShow(Sender: TObject);
begin
    cmbxSourceAlias.SetFocus;
end;

end.
```

```
Number of records applied: '+IntToStr(BatchMove1.MovedCount),mtInformation,[mbOK],0);
end;

<------->procedure TForm1.chkbxAbortKeyClick(Sender: TObject);
begin
  BatchMove1.AbortOnKeyViol := chkbxAbortKey.Checked;
end;

<------->procedure TForm1.chkbxAbortProblemClick(Sender: TObject);
begin
  BatchMove1.AbortOnProblem := chkbxAbortProblem.Checked;
end;

<------->procedure TForm1.chkbxTransClick(Sender: TObject);
begin
  BatchMove1.Transliterate := chkbxTrans.Checked;
end;

function TForm1.IsStringsEqual(const s1,s2 : string): boolean;
begin
   Result := UpperCase(s1) = UpperCase(s2);
end;


// set the batch mode
<------->procedure TForm1.cmbxModeChange(Sender: TObject);
begin
  if cmbxMode.ItemIndex <> -1 then
  begin
    if IsStringsEqual(cmbxMode.Items[cmbxMode.ItemIndex],'Append') then
      BatchMove1.Mode := batAppend
```

Objects36
Object$
الملف النصي لبـرنـامج
(BatchMv)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١
<------->object Form1: TForm1
Left = 173
Top = 120
Width = 766
Height = 584
Caption = 'TBatchMove Example'
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = False
Position = poDesktopCenter
OnCreate = FormCreate
OnShow = FormShow
PixelsPerInch = 96
TextHeight = 13
<------->object Bevel1: TBevel
Left = 312
Top = 8
Width = 265
Height = 169
Shape = bsFrame
Style = bsRaised
end

<------->object Label6: TLabel
Left = 344
Top = 48
Width = 25
Height = 13
Caption = 'Alias:'
end
<------->object Label7: TLabel
Left = 324
Top = 88
Width = 61
Height = 13
Caption = 'Table Name:'
end
<------->object Label8: TLabel
Left = 348
Top = 128
Width = 29
Height = 13
Caption = 'Index:'
end
<------->object Bevel2: TBevel
Left = 16
Top = 8
Width = 265
Height = 169
Shape = bsFrame
Style = bsRaised
end
<------->object Label5: TLabel
Left = 48
Top = 128
Width = 29
Height = 13
Caption = 'Index:'
end
<------->object Label4: TLabel
Left = 24
Top = 88
Width = 61
Height = 13
Caption = 'Table Name:'
end
<------->object Label3: TLabel
Left = 48
Top = 48
Width = 25
Height = 13

Font.Height = -12
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
end
<------->object Label9: TLabel
Left = 28
Top = 336
Width = 61
Height = 13
Caption = 'Batch Mode:'
end
<------->object Label12: TLabel
Left = 288
Top = 232
Width = 110
Height = 13
Caption = 'Changed Table Name: '
end
<------->object Label13: TLabel
Left = 272
Top = 272
Width = 128
Height = 13
Caption = 'Key Violation Table Name: '
end
<------->object Label14: TLabel
Left = 296
Top = 312
Width = 105
Height = 13
Caption = 'Problem Table Name: '
end
<------->object Label11: TLabel
Left = 256
Top = 352
Width = 145
Height = 13
Caption = 'Record Count (Leave 0 for all):'
end
<------->object cmbxDestAlias: TComboBox
Left = 400
Top = 48
Width = 145

Caption = 'Alias:'
end
<------->object Bevel3: TBevel
Left = 16
Top = 192
Width = 561
Height = 201
Shape = bsFrame
Style = bsRaised
end
<------->object Label1: TLabel
Left = 104
Top = 16
Width = 77
Height = 13
Caption = 'Source Table'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -12
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
end
<------->object Label2: TLabel
Left = 400
Top = 16
Width = 101
Height = 13
Caption = 'Destination Table'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -12
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
end
<------->object Label10: TLabel
Left = 224
Top = 200
Width = 131
Height = 13
Caption = 'Options/Error Handling'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText

```
  Left = 96
  Top = 88
  Width = 145
  Height = 21
  Style = csDropDownList
  ItemHeight = 13
  Sorted = True
  TabOrder = 4
  OnChange =
cmbxSourceTableChange
 end
 <------->object cmbxSourceIndex:
TComboBox
  Left = 96
  Top = 128
  Width = 145
  Height = 21
  Style = csDropDownList
  ItemHeight = 13
  Sorted = True
  TabOrder = 5
  OnChange =
cmbxSourceIndexChange
 end
 <------->object cmbxMode:
TComboBox
  Left = 96
  Top = 336
  Width = 145
  Height = 21
  Style = csDropDownList
  ItemHeight = 13
  Sorted = True
  TabOrder = 6
  OnChange = cmbxModeChange
  Items.Strings = (
   'Append'
   'Append Update'
   'Copy'
   'Delete'
   'Update')
 end
 <------->object Button1: TButton
  Left = 230
  Top = 404
  Width = 131
  Height = 25
  Caption = 'Execute BatchMove'
```

```
  Height = 21
  Style = csDropDownList
  ItemHeight = 13
  Sorted = True
  TabOrder = 0
  OnChange =
cmbxDestAliasChange
 end
 <------->object cmbxDestTable:
TComboBox
  Left = 400
  Top = 88
  Width = 145
  Height = 21
  ItemHeight = 13
  Sorted = True
  TabOrder = 1
  OnChange =
cmbxDestTableChange
 end
 <------->object cmbxDestIndex:
TComboBox
  Left = 400
  Top = 128
  Width = 145
  Height = 21
  Style = csDropDownList
  ItemHeight = 13
  Sorted = True
  TabOrder = 2
  OnChange =
cmbxDestIndexChange
 end
 <------->object cmbxSourceAlias:
TComboBox
  Left = 96
  Top = 48
  Width = 145
  Height = 21
  Style = csDropDownList
  ItemHeight = 13
  Sorted = True
  TabOrder = 3
  OnChange =
cmbxSourceAliasChange
 end
 <------->object cmbxSourceTable:
TComboBox
```

TabOrder = 11
end
<------->object edtKeyVioTbl: TEdit
Left = 416
Top = 272
Width = 121
Height = 21
TabOrder = 12
end
<------->object edtProbTbl: TEdit
Left = 416
Top = 312
Width = 121
Height = 21
TabOrder = 13
end
<------->object edtRecCount: TEdit
Left = 416
Top = 352
Width = 121
Height = 21
TabOrder = 14
Text = '0'
OnKeyPress = edtRecCountKeyPress
end
<------->object BatchMove1: TBatchMove
Destination = tblDest
Source = tblSource
Transliterate = False
Left = 280
Top = 88
end
<------->object tblSource: TTable
Left = 280
Top = 24
end
<------->object tblDest: TTable
Left = 280
Top = 144
end
End

TabOrder = 7
OnClick = Button1Click
end
<------->object chkbxAbortKey: TCheckBox
Left = 32
Top = 232
Width = 193
Height = 17
Caption = 'Abort on key violation'
Checked = True
State = cbChecked
TabOrder = 8
OnClick = chkbxAbortKeyClick
end
<------->object chkbxAbortProblem: TCheckBox
Left = 32
Top = 264
Width = 193
Height = 17
Caption = 'Abort on problem (data is truncated)'
Checked = True
State = cbChecked
TabOrder = 9
OnClick = chkbxAbortProblemClick
end
<------->object chkbxTrans: TCheckBox
Left = 32
Top = 296
Width = 193
Height = 17
Caption = 'Transliterate Characters'
Checked = True
State = cbChecked
TabOrder = 10
OnClick = chkbxTransClick
end
<------->object edtChangedTable: TEdit
Left = 416
Top = 232
Width = 121
Height = 21

ObjectS

الملف النصي لبرنامج

(BatchMv2)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
object FormMain: TFormMain
  Left = 156
  Top = 110
  Width = 766
  Height = 584
  Caption = 'TBatchMove Example'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  Position = poDesktopCenter
  OnCreate = FormCreate
  OnShow = FormShow
  PixelsPerInch = 96
  TextHeight = 13
  object Bevel1: TBevel
    Left = 312
    Top = 8
    Width = 265
    Height = 169
    Shape = bsFrame
    Style = bsRaised
  end
  object Label6: TLabel
    Left = 344
    Top = 48
    Width = 25
    Height = 13
    Caption = 'Alias:'
  end
  object Label7: TLabel
    Left = 324
    Top = 88
    Width = 61
    Height = 13
    Caption = 'Table Name:'
  end
  object Label8: TLabel
    Left = 348
    Top = 128
    Width = 29
    Height = 13
    Caption = 'Index:'
  end
  object Bevel2: TBevel
    Left = 16
    Top = 8
    Width = 265
    Height = 169
    Shape = bsFrame
    Style = bsRaised
  end
  object Label5: TLabel
    Left = 48
    Top = 128
    Width = 29
    Height = 13
    Caption = 'Index:'
  end
  object Label4: TLabel
    Left = 24
    Top = 88
    Width = 61
    Height = 13
    Caption = 'Table Name:'
  end
  object Label3: TLabel
    Left = 48
    Top = 48
    Width = 25
    Height = 13
    Caption = 'Alias:'
  end
  object Bevel3: TBevel
    Left = 16
    Top = 192
    Width = 561
```

```
    Left = 28
    Top = 336
    Width = 61
    Height = 13
    Caption = 'Batch Mode:'
  end
  object Label12: TLabel
    Left = 288
    Top = 232
    Width = 110
    Height = 13
    Caption = 'Changed Table Name: '
  end
  object Label13: TLabel
    Left = 272
    Top = 272
    Width = 128
    Height = 13
    Caption = 'Key Violation Table Name: '
  end
  object Label14: TLabel
    Left = 296
    Top = 312
    Width = 105
    Height = 13
    Caption = 'Problem Table Name: '
  end
  object Label11: TLabel
    Left = 256
    Top = 352
    Width = 145
    Height = 13
    Caption = 'Record Count (Leave 0 for all):'
  end
  object cmbxDestAlias: TComboBox
    Left = 400
    Top = 48
    Width = 145
    Height = 21
    Style = csDropDownList
    ItemHeight = 13
    Sorted = True
    TabOrder = 0
    OnChange = cmbxDestAliasChange
```

```
    Height = 201
    Shape = bsFrame
    Style = bsRaised
  end
  object Label1: TLabel
    Left = 104
    Top = 16
    Width = 77
    Height = 13
    Caption = 'Source Table'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -12
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentFont = False
  end
  object Label2: TLabel
    Left = 400
    Top = 16
    Width = 101
    Height = 13
    Caption = 'Destination Table'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -12
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentFont = False
  end
  object Label10: TLabel
    Left = 224
    Top = 200
    Width = 131
    Height = 13
    Caption = 'Options/Error Handling'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -12
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentFont = False
  end
  object Label9: TLabel
```

```
    TabOrder = 4
    OnChange = cmbxSourceTableChange
  end
  object cmbxSourceIndex: TComboBox
    Left = 96
    Top = 128
    Width = 145
    Height = 21
    Style = csDropDownList
    ItemHeight = 13
    Sorted = True
    TabOrder = 5
    OnChange = cmbxSourceIndexChange
  end
  object cmbxMode: TComboBox
    Left = 96
    Top = 336
    Width = 145
    Height = 21
    Style = csDropDownList
    ItemHeight = 13
    Sorted = True
    TabOrder = 6
    OnChange = cmbxModeChange
    Items.Strings = (
      'Append'
      'Append Update'
      'Copy'
      'Delete'
      'Update')
  end
  object Button1: TButton
    Left = 230
    Top = 404
    Width = 131
    Height = 25
    Caption = 'Execute BatchMove'
    TabOrder = 7
    OnClick = Button1Click
  end
  object chkbxAbortKey: TCheckBox
    Left = 32
    Top = 232
    Width = 193
  end
  object cmbxDestTable: TComboBox
    Left = 400
    Top = 88
    Width = 145
    Height = 21
    ItemHeight = 13
    Sorted = True
    TabOrder = 1
    OnChange = cmbxDestTableChange
  end
  object cmbxDestIndex: TComboBox
    Left = 400
    Top = 128
    Width = 145
    Height = 21
    Style = csDropDownList
    ItemHeight = 13
    Sorted = True
    TabOrder = 2
    OnChange = cmbxDestIndexChange
  end
  object cmbxSourceAlias: TComboBox
    Left = 96
    Top = 48
    Width = 145
    Height = 21
    Style = csDropDownList
    ItemHeight = 13
    Sorted = True
    TabOrder = 3
    OnChange = cmbxSourceAliasChange
  end
  object cmbxSourceTable: TComboBox
    Left = 96
    Top = 88
    Width = 145
    Height = 21
    Style = csDropDownList
    ItemHeight = 13
    Sorted = True
```

```
    Left = 416
    Top = 312
    Width = 121
    Height = 24
    TabOrder = 13
  end
  object edtRecCount: TEdit
    Left = 416
    Top = 352
    Width = 121
    Height = 24
    TabOrder = 14
    Text = '0'
    OnKeyPress =
edtRecCountKeyPress
  end
  object BatchMove1: TBatchMove
    Destination = tblDest
    Source = tblSource
    Transliterate = False
    Left = 280
    Top = 88
  end
  object tblSource: TTable
    Left = 280
    Top = 24
  end
  object tblDest: TTable
    Left = 280
    Top = 144
  end
End
```

Object$
الملف التنفيذي لبرنامج
(BatchMv2)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
unit Main;
```

```
    Height = 17
    Caption = 'Abort on key violation'
    Checked = True
    State = cbChecked
    TabOrder = 8
    OnClick = chkbxAbortKeyClick
  end
  object chkbxAbortProblem:
TCheckBox
    Left = 32
    Top = 264
    Width = 193
    Height = 17
    Caption = 'Abort on problem
(data is truncated)'
    Checked = True
    State = cbChecked
    TabOrder = 9
    OnClick =
chkbxAbortProblemClick
  end
  object chkbxTrans: TCheckBox
    Left = 32
    Top = 296
    Width = 193
    Height = 17
    Caption = 'Transliterate
Characters'
    Checked = True
    State = cbChecked
    TabOrder = 10
    OnClick = chkbxTransClick
  end
  object edtChangedTable: TEdit
    Left = 416
    Top = 232
    Width = 121
    Height = 24
    TabOrder = 11
  end
  object edtKeyVioTbl: TEdit
    Left = 416
    Top = 272
    Width = 121
    Height = 24
    TabOrder = 12
  end
  object edtProbTbl: TEdit
```

```
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls, Db, DBTables, ComCtrls;

type
  TFormMain = class(TForm)
    BatchMove1: TBatchMove;
    tblSource: TTable;
    tblDest: TTable;
    Bevel1: TBevel ;
    Label6: TLabel;
    cmbxDestAlias: TComboBox;
    Label7: TLabel;
    cmbxDestTable: TComboBox;
    Label8: TLabel;
    cmbxDestIndex: TComboBox;
    Bevel2: TBevel;
    cmbxSourceAlias: TComboBox;
    cmbxSourceTable: TComboBox;
    cmbxSourceIndex: TComboBox;
    Label5: TLabel;
    Label4: TLabel;
    Label3: TLabel;
    cmbxMode: TComboBox;
    Button1: TButton;
    Bevel3: TBevel;
    chkbxAbortKey: TCheckBox;
    chkbxAbortProblem: TCheckBox;
    chkbxTrans: TCheckBox;
    Label1: TLabel;
    Label2: TLabel;
    Label10: TLabel;
    Label9: TLabel;
    Label12: TLabel;
    edtChangedTable: TEdit;
    Label13: TLabel;
    edtKeyVioTbl: TEdit;
    Label14: TLabel;
    edtProbTbl: TEdit;
    Label11: TLabel;
    edtRecCount: TEdit;
    procedure FormCreate(Sender: TObject);
    procedure cmbxSourceAliasChange(Sender: TObject);
    procedure cmbxDestAliasChange(Sender: TObject);
    procedure cmbxSourceTableChange(Sender: TObject);
    procedure cmbxDestTableChange(Sender: TObject);
    procedure cmbxSourceIndexChange(Sender: TObject);
    procedure cmbxDestIndexChange(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure chkbxAbortKeyClick(Sender: TObject);
    procedure chkbxAbortProblemClick(Sender: TObject);
    procedure chkbxTransClick(Sender: TObject);
    procedure cmbxModeChange(Sender: TObject);
    procedure edtRecCountKeyPress(Sender: TObject; var Key: Char);
    procedure FormShow(Sender: TObject);
  private
    { Private declarations }
    function IsStringsEqual(const s1,s2 : string): boolean;
  public
    { Public declarations }
  end;

var
  FormMain: TFormMain;

implementation
```

```
{$R *.DFM}

resourcestring
  sIncomplete = 'Incomplete input.';
  sCompleted = 'BatchMove complete. Number of records applied: ';

procedure TFormMain.FormCreate(Sender: TObject); // Get alias names
begin
  if Session.Active = false then
    Session.Open;
  Session.GetAliasNames(CmbxSourceAlias.Items);
  cmbxDestAlias.Items := cmbxSourceAlias.Items;
end;

procedure TFormMain.cmbxSourceAliasChange(Sender: TObject);
begin
  if cmbxSourceAlias.ItemIndex <> -1 then
  begin
    tblSource.DatabaseName := cmbxSourceAlias.Items[cmbxSourceAlias.ItemIndex];
    Session.GetTableNames(tblSource.DatabaseName,'',true,false,cmbxSourceTable.Items);
  end
  else
  begin
    tblSource.DatabaseName := '';
    cmbxSourceTable.Items.Clear;
  end;
end;

procedure TFormMain.cmbxDestAliasChange(Sender: TObject);
begin
  if cmbxDestAlias.ItemIndex <> -1 then
  begin
    tblDest.DatabaseName := cmbxDestAlias.Items[cmbxDestAlias.ItemIndex];
    Session.GetTableNames(tblDest.DatabaseName,'',true,false,cmbxDestTable.Items);
  end
  else
  begin
    tblDest.DatabaseName := '';
    cmbxDestTable.Items.Clear;
  end;
end;

procedure TFormMain.cmbxSourceTableChange(Sender: TObject);
begin
  if cmbxSourceTable.ItemIndex <> -1 then
  begin
    tblSource.TableName := cmbxSourceTable.Items[cmbxSourceTable.ItemIndex];
    tblSource.GetIndexNames(cmbxSourceIndex.Items);
  end
  else
  begin
    tblSource.TableName := '';
    cmbxSourceIndex.Items.Clear;
  end;

end;

procedure TFormMain.cmbxDestTableChange(Sender: TObject);
begin
  if cmbxDestTable.ItemIndex <> -1 then
  begin
    tblDest.TableName := cmbxDestTable.Items[cmbxDestTable.ItemIndex];
```

```
  if tblDest.TableName = '' then
    tblDest.TableName := cmbxDestTable.Text;
  if ((tblSource.DatabaseName <> '') and // test for enough input
     (tblSource.TableName <> '') and
     (tblDest.DatabaseName <> '') and
     (tblDest.TableName <> '') and
     (cmbxMode.items[cmbxMode.ItemIndex] <> '')) then
  begin
    BatchMove1.ChangedTableName := edtChangedTable.Text;  // more batchmove setup
    BatchMove1.KeyViolTableName := edtKeyVioTbl.Text;
    BatchMove1.ProblemTableName := edtProbTbl.Text;
    BatchMove1.RecordCount := StrToInt(edtRecCount.Text);
  end
  else
  begin
    MessageDlg(sIncomplete, mtError, [mbOK], 0);
    exit;
  end;
  BatchMove1.Execute;  // run the batchmove
  MessageDlg(sCompleted + IntToStr(BatchMove1.MovedCount),mtInformation,[mbOK],0);
end;

procedure TFormMain.chkbxAbortKeyClick(Sender: TObject);
begin
  BatchMove1.AbortOnKeyViol := chkbxAbortKey.Checked;
end;

procedure TFormMain.chkbxAbortProblemClick(Sender: TObject);
begin
  BatchMove1.AbortOnProblem := chkbxAbortProblem.Checked;
tblDest.GetIndexNames(cmbxDestIndex.Items);
  end
  else
  begin
    tblDest.TableName := '';
    cmbxDestIndex.Items.Clear;
  end;
end;

procedure TFormMain.cmbxSourceIndexChange(Sender: TObject);
begin
  if cmbxSourceIndex.ItemIndex <> -1 then
  begin
    tblSource.IndexName := cmbxSourceIndex.Items[cmbxSourceIndex.ItemIndex];
  end
  else
  begin
     tblSource.IndexName := '';
  end;
end;

procedure TFormMain.cmbxDestIndexChange(Sender: TObject);
begin
  if cmbxDestIndex.ItemIndex <> -1 then
  begin
    tblDest.IndexName := cmbxDestIndex.Items[cmbxDestIndex.ItemIndex];
  end
  else
  begin
     tblDest.IndexName := '';
  end;

end;

procedure TFormMain.Button1Click(Sender: TObject);
begin
```

```
    MessageDlg('Batch mode not found',mtError,[mbOK],0);
  end;
end;


// only allow numbers to be typed in
procedure TFormMain.edtRecCountKeyPress(Sender: TObject; var Key: Char);
begin
  if ((key in ['0'..'9'] = false) and (word(key) <> VK_BACK)) then
    key := #0;
end;

procedure TFormMain.FormShow(Sender: TObject);
begin
  cmbxSourceAlias.SetFocus;
end;

end.

end;

procedure TFormMain.chkbxTransClick(Sender: TObject);
begin
  BatchMove1.Transliterate := chkbxTrans.Checked;
end;

function TFormMain.IsStringsEqual(const s1,s2 : string): boolean;
begin
  Result := UpperCase(s1) = UpperCase(s2);
end;


// set the batch mode
procedure TFormMain.cmbxModeChange(Sender: TObject);
begin
  if cmbxMode.ItemIndex <> -1 then
  begin
    if IsStringsEqual(cmbxMode.Items[cmbxMode.ItemIndex],'Append') then
      BatchMove1.Mode := batAppend
    else if IsStringsEqual(cmbxMode.Items[cmbxMode.ItemIndex],'Copy') then
      BatchMove1.Mode := batCopy
    else if IsStringsEqual(cmbxMode.Items[cmbxMode.ItemIndex],'Append Update') then
      BatchMove1.Mode := batAppendUpdate
    else if IsStringsEqual(cmbxMode.Items[cmbxMode.ItemIndex],'Delete') then
      BatchMove1.Mode := batDelete
    else if IsStringsEqual(cmbxMode.Items[cmbxMode.ItemIndex],'Update') then
      BatchMove1.Mode := batUpdate
    else
```

Objects
الملف التنفيذي لبرنامج
(Scroll bar)
إعداد **علاء الدين اللباد**
هاتف ٠٩٤٤٥٧٥٣٧١
الملف التنفيذي للبرنامج
program scroll;
وفيه
Procedures 38
program scroll;
uses
Forms,
scroldem in 'scroldem.pas' {ScrollForm1},
about in 'about.pas' {AboutBox};
{$R *.RES}
begin
Application.Initialize;
Application.CreateForm(TScrollForm1, ScrollForm1);
Application.Run;
end.
//
// this example demonstrates the use of TScrollBar, including how to
// properly set scrolling ranges and how to respond to the scroll bar
// notification events. in addition, an interface is provided to the
// user to adjust the values to see how the different parameters affect

```
// the scrolling behavior.  no attempt is made to verify that the parameters
// supplied by the user make any sense.
//
unit scroldem;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Menus, StdCtrls, ComCtrls, ExtCtrls;

type
  TScrollForm1 = class(TForm)
    Bevel1: TBevel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label1: TLabel;
    Vertical: TLabel;
    ImgPanel: TPanel;
    Image: TImage;
    HScrollb: TScrollBar;
    VScrollb: TScrollBar;
    HUnits: TEdit;
    HPage: TEdit;
    HMax: TEdit;
    VUnits: TEdit;
    VPage: TEdit;
    VMax: TEdit;
    HUnitsUpDown: TUpDown;
    HPageUpDown: TUpDown;
    HMaxUpDown: TUpDown;
    VUnitsUpDown: TUpDown;
    VMaxUpDown: TUpDown;
    VPageUpDown: TUpDown;
    DefaultBtn: TButton;
    ApplyBtn: TButton;
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    Open1: TMenuItem;
    N1: TMenuItem;
    Exit1: TMenuItem;
    View1: TMenuItem;
```

```
    VScrollMenu: TMenuItem;
    HScrollMenu: TMenuItem;
    Help1: TMenuItem;
    About1: TMenuItem;
    OpenDialog: TOpenDialog;
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure Exit1Click(Sender: TObject);
    <-------> procedure View1Click(Sender: TObject);
    <-------> procedure VScrollMenuClick(Sender: TObject);
    <-------> procedure HScrollMenuClick(Sender: TObject);
    <-------> procedure Open1Click(Sender: TObject);
    <-------> procedure VScrollbChange(Sender: TObject);
    <-------> procedure HScrollbScroll(Sender: TObject; ScrollCode: TScrollCode; var ScrollPos: Integer);
    <-------> procedure ImgPanelResize(Sender: TObject);
    <-------> procedure About1Click(Sender: TObject);
    <-------> procedure ApplyBtnClick(Sender: TObject);
    <-------> procedure DefaultBtnClick(Sender: TObject);
    <-------> procedure UpDownClick(Sender: TObject; Button: TUDBtnType);

  private
    { Private declarations }
    Units: TPoint;
    FDirty: Boolean;

    <-------> procedure SetDirty(d: Boolean);
    <-------> procedure ImageLoad(filename: string);
    <-------> procedure ScrollAdjust(update: Boolean);
    <-------> procedure ScrollReset;
    <-------> procedure UpdateDisplay;
    property Dirty: Boolean read FDirty write SetDirty;

  public
    //constructor Create(Owner: TComponent); override;
    { Public declarations }
  end;

var
  ScrollForm1: TScrollForm1;

implementation

uses about;


{$R *.DFM}

const
  DEF_SCROLL_UNITS = 8;

<-------> procedure TScrollForm1.FormCreate(Sender: TObject);
begin
  ScrollReset;
end;

<-------> procedure TScrollForm1.Exit1Click(Sender: TObject);
begin
  Close;
end;

<-------> procedure TScrollForm1.View1Click(Sender: TObject);
begin
  HScrollMenu.Checked := HScrollb.Visible;
  VScrollMenu.Checked := VScrollb.Visible;
end;

<-------> procedure
```

```
needed for a few reasons:
//   1) you want to do something special with particular scroll
//      events
//   2) the range of your scrollbar is larger than [0, 65535].  in
//      this case, the ScrollPos element is invalid and much of the
//      scrolling has to be done manually.
//
<-------> procedure TScrollForm1.HScrollbScroll(Sender: TObject; ScrollCode: TScrollCode; var ScrollPos: Integer);
begin
  if HScrollb.Max <= 65535 then
    Image.Left := -Units.x * ScrollPos
  else
    ShowMessage('Scrollbar ranges > 65535 not supported');
end;

//
// The panel's size doesn't change in this example, so
// this event will only happen once.  In general, the range
// of the scrollbars will need to be changed when the size of
// the visible window is changed.
//
<-------> procedure TScrollForm1.ImgPanelResize(Sender: TObject);
begin
  ScrollAdjust(True);
end;

function scale(n, u: Integer): Integer;
begin
  Result := (n + u - 1) div u;
end;

function maxval(a, b: Integer): Integer;
begin
TScrollForm1.VScrollMenuClick(Sender: TObject);
begin
  VScrollb.Visible := not VScrollb.Visible;
end;

<-------> procedure TScrollForm1.HScrollMenuClick(Sender: TObject);
begin
  HScrollb.Visible := not HScrollb.Visible;
end;

<-------> procedure TScrollForm1.Open1Click(Sender: TObject);
begin
  if OpenDialog.Execute then
    ImageLoad(OpenDialog.FileName);
end;

//
// Hooking the OnChange event is the simplest way to handle the
// scrollbar notification.  Note the check to ensure that the
// that the sender is the vertical scrollbar.  While irrelevant for
// this example, some forms may need to distinguish between various
// scrollbars.
//
<-------> procedure TScrollForm1.VScrollbChange(Sender: TObject);
begin
  if Sender as TScrollBar = VScrollb then
    Image.Top := -Units.y * VScrollb.Position;
end;

//
// Hooking the OnScroll event is another option for handling the
// scrollbar notification.  This is
```

```
// is ignored.
//
<-------> procedure TScrollForm1.ScrollReset;
begin
  // this fixed size works nice, although in some cases basing
  // the scroll units upon the size of the image works well, too.
  Units.x := DEF_SCROLL_UNITS;
  Units.y := DEF_SCROLL_UNITS;

  if Image.Picture <> nil then
  begin
    // move the image and the scrollbars to their "home" location
    Image.Top := 0;
    Image.Left := 0;
    HScrollb.Position := 0;
    VScrollb.Position := 0;

    // negative scrolling ranges aren't worth the trouble, so set
    // the range to go from [0, M].  See 'ScrollAdjust' to see how M
    // is calculated.
    HScrollb.Min := 0;
    VScrollb.Min := 0;
    ScrollAdjust(False);

    HScrollb.LargeChange := scale(HScrollb.Max, Units.x);
    VScrollb.LargeChange := scale(VScrollb.Max, Units.y);

    HScrollb.Visible := True;
    VScrollb.Visible := True;
  end;

  UpdateDisplay;
end;


<-------> procedure TScrollForm1.SetDirty(d: Boolean);
begin
  if d <> FDirty then
  begin
  if a >= b then
    Result := a
  else
    Result := b;
end;

//
// Readjust the range of the scrollbars to match the new size
// of the visible window.  (Also called by ScrollReset to avoid
// duplicating the code).
//
<-------> procedure TScrollForm1.ScrollAdjust(update: Boolean);
begin
  // set the maximum scrolling to be large enough to scroll the whole
  // image, but not so much that it can scroll off the window.  we
  // divide by the number of Units (>1) because scrolling one pixel at
  // a time is way too slow for most images of any size.

  if Image.Picture <> nil then
  begin
    HScrollb.Max := maxval(0, scale(Image.Width - ImgPanel.Width, Units.x));
    VScrollb.Max := maxval(0, scale(Image.Height - ImgPanel.Height, Units.y));
  end;

  if update then
    UpdateDisplay;
end;

//
// ScrollReset() is used to put the scrollbars into a sane "initial"
// state.  this needs to be done each time the image is changed,
// since the amount we scroll depends upon the image size.  for
// simplicity, the special case of "new image size" == "old image size"
```

```
<-------> procedure
TScrollForm1.DefaultBtnClick(Sender: TObject);
begin
  ScrollReset;
  Dirty := False;
end;

<-------> procedure
TScrollForm1.UpDownClick(Sender
: TObject; Button: TUDBtnType);
begin
  Dirty := True;
end;

<-------> procedure
TScrollForm1.UpdateDisplay;
begin
  HUnitsUpDown.Position := Units.x;
  HPageUpDown.Position :=
HScrollb.LargeChange;
  HMaxUpDown.Position :=
HScrollb.Max;

  VUnitsUpDown.Position := Units.y;
  VPageUpDown.Position :=
VScrollb.LargeChange;
  VMaxUpDown.Position :=
VScrollb.Max;

  Dirty := False;
end;

end.



unit about;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls;

type
```

```
    FDirty := d;
    ApplyBtn.Enabled := FDirty;
  end;
end;


<-------> procedure
TScrollForm1.ImageLoad(filename:
string);
begin

Image.Picture.LoadFromFile(filena
me);
  ScrollReset;
end;

<-------> procedure
TScrollForm1.About1Click(Sender:
TObject);
var
  about: TAboutBox;
begin
  about := TAboutBox.Create(Self);
  try
    about.ShowModal;
  finally
    about.Free;
  end;
end;


<-------> procedure
TScrollForm1.ApplyBtnClick(Sende
r: TObject);
begin
  Units.x := HUnitsUpDown.Position;
  HScrollb.LargeChange :=
HPageUpDown.Position;
  HScrollb.Max :=
HMaxUpDown.Position;

  Units.y := VUnitsUpDown.Position;
  VScrollb.LargeChange :=
VPageUpDown.Position;
  VScrollb.Max :=
VMaxUpDown.Position;

  Dirty := False;
end;
```

```
biMinimize]
  BorderStyle = bsSingle
  Caption = 'ScrollDemo'
  ClientHeight = 194
  ClientWidth = 436
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  Menu = MainMenu1
  OldCreateOrder = True
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Bevel1: TBevel
    Left = 0
    Top = 0
    Width = 436
    Height = 9
    Align = alTop
    Shape = bsTopLine
  end
  <-------> object Label3: TLabel
    Left = 238
    Top = 42
    Width = 24
    Height = 13
    Caption = 'Units'
  end
  <-------> object Label4: TLabel
    Left = 198
    Top = 78
    Width = 64
    Height = 13
    Caption = 'LargeChange'
  end
  <-------> object Label5: TLabel
    Left = 242
    Top = 117
    Width = 20
    Height = 13
    Caption = 'Max'
  end
  <-------> object Label1: TLabel
    Left = 280
```

```
  TAboutBox = class(TForm)
    Button1: TButton;
    Label1: TLabel;
    Label2: TLabel;
    <-------> procedure
Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  AboutBox: TAboutBox;

implementation

{$R *.DFM}

<-------> procedure
TAboutBox.Button1Click(Sender:
TObject);
begin
  Close;
end;

end.
```

الملف النصي للبرنامج

Scroll bar

وفيه

Objects 40

من إعداد

علاء الدين اللباد

٠٩٤٤٥٧٥٣٧١

```
<-------> object ScrollForm1:
TScrollForm1
  Left = 181
  Top = 156
  BorderIcons = [biSystemMenu,
```

```
    ReadOnly = True
    TabOrder = 4
    Text = '0'
  end
  <-------> object HMax: TEdit
    Left = 276
    Top = 113
    Width = 48
    Height = 21
    ReadOnly = True
    TabOrder = 5
    Text = '0'
  end
  <-------> object VUnits: TEdit
    Left = 363
    Top = 38
    Width = 48
    Height = 21
    ReadOnly = True
    TabOrder = 6
    Text = '1'
  end
  <-------> object VPage: TEdit
    Left = 363
    Top = 74
    Width = 48
    Height = 21
    ReadOnly = True
    TabOrder = 7
    Text = '0'
  end
  <-------> object VMax: TEdit
    Left = 363
    Top = 113
    Width = 48
    Height = 21
    ReadOnly = True
    TabOrder = 8
    Text = '0'
  end
  <-------> object HUnitsUpDown:
TUpDown
    Left = 323
    Top = 38
    Width = 15
    Height = 21
    Associate = HUnits
    Min = 1
```

```
    Top = 16
    Width = 47
    Height = 13
    Caption = 'Horizontal'
  end
  <-------> object Vertical: TLabel
    Left = 376
    Top = 16
    Width = 35
    Height = 13
    Caption = 'Vertical'
  end
  <-------> object HScrollb:
TScrollBar
    Left = 9
    Top = 169
    Width = 159
    Height = 16
    Max = 32767
    PageSize = 0
    TabOrder = 1
    OnScroll = HScrollbScroll
  end
  <-------> object VScrollb:
TScrollBar
    Left = 169
    Top = 9
    Width = 16
    Height = 157
    Kind = sbVertical
    PageSize = 0
    TabOrder = 2
    OnChange = VScrollbChange
  end
  <-------> object HUnits: TEdit
    Left = 275
    Top = 38
    Width = 48
    Height = 21
    ReadOnly = True
    TabOrder = 3
    Text = '1'
  end
  <-------> object HPage: TEdit
    Left = 275
    Top = 74
    Width = 48
    Height = 21
```

TabOrder = 13
OnClick = UpDownClick
end
<-------> object VPageUpDown: TUpDown
Left = 411
Top = 74
Width = 15
Height = 21
Associate = VPage
Max = 32767
TabOrder = 14
OnClick = UpDownClick
end
<-------> object DefaultBtn: TButton
Left = 348
Top = 157
Width = 75
Height = 25
Caption = '&Defaults'
ParentShowHint = False
ShowHint = False
TabOrder = 15
OnClick = DefaultBtnClick
end
<-------> object ApplyBtn: TButton
Left = 265
Top = 157
Width = 75
Height = 25
Caption = '&Apply'
Enabled = False
TabOrder = 16
OnClick = ApplyBtnClick
end
<-------> object ImgPanel: TPanel
Left = 8
Top = 8
Width = 160
Height = 160
BevelOuter = bvNone
BorderStyle = bsSingle
TabOrder = 0
<-------> object Image: TImage
Left = 0
Top = 0
Width = 600

Max = 32767
Position = 1
TabOrder = 9
OnClick = UpDownClick
end
<-------> object HPageUpDown: TUpDown
Left = 323
Top = 74
Width = 15
Height = 21
Associate = HPage
Max = 32767
TabOrder = 10
OnClick = UpDownClick
end
<-------> object HMaxUpDown: TUpDown
Left = 324
Top = 113
Width = 15
Height = 21
Associate = HMax
Max = 32767
TabOrder = 11
OnClick = UpDownClick
end
<-------> object VUnitsUpDown: TUpDown
Left = 411
Top = 38
Width = 15
Height = 21
Associate = VUnits
Min = 1
Max = 32767
Position = 1
TabOrder = 12
OnClick = UpDownClick
end
<-------> object VMaxUpDown: TUpDown
Left = 411
Top = 113
Width = 15
Height = 21
Associate = VMax
Max = 32767

```
    end
   end
  end
  <-------> object OpenDialog: TOpenDialog
    DefaultExt = '.bmp'
    Filter = 
      'Images (*.bmp;*.ico)|*.bmp;*.ico|Bitmaps (*.bmp)|*.bmp|Icons (*.' +
      'ico)|*.ico'
    Options = [ofHideReadOnly, ofPathMustExist, ofFileMustExist]
    Left = 230
    Top = 158
  end
End

<-------> object AboutBox: TAboutBox
  Left = 200
  Top = 108
  BorderIcons = [biSystemMenu]
  BorderStyle = bsDialog
  Caption = 'About ScrollDemo'
  ClientHeight = 135
  ClientWidth = 239
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Label1: TLabel
    Left = 24
    Top = 24
    Width = 191
    Height = 20
    Caption = 'ScrollBar Demonstation'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
```

```
      Height = 470
      AutoSize = True
      Picture.Data = {
        }
    end
  end
  <-------> object MainMenu1: TMainMenu
    Left = 196
    Top = 159
    <-------> object File1: TMenuItem
      Caption = '&File'
      <-------> object Open1: TMenuItem
        Caption = '&Open...'
        OnClick = Open1Click
      end
      <-------> object N1: TMenuItem
        Caption = '-'
      end
      <-------> object Exit1: TMenuItem
        Caption = 'E&xit'
        OnClick = Exit1Click
      end
    end
    <-------> object View1: TMenuItem
      Caption = '&Scroll'
      OnClick = View1Click
      <-------> object VScrollMenu: TMenuItem
        Caption = '&Vertical'
        OnClick = VScrollMenuClick
      end
      <-------> object HScrollMenu: TMenuItem
        Caption = '&Horizontal'
        OnClick = HScrollMenuClick
      end
    end
    <-------> object Help1: TMenuItem
      Caption = '&Help'
      <-------> object About1: TMenuItem
        Caption = '&About...'
        OnClick = About1Click
```

type
TMainForm = class(TForm)
ApplicationEvents: TApplicationEvents;
ActionList: TActionList;
Action: TAction;
MainMenu: TMainMenu;
MenuAction: TMenuItem;
MenuException: TMenuItem;
HintButton: TButton;
lbOnMessage: TListBox;
lblOnMessage: TLabel;
lbOther: TListBox;
lblOther: TLabel;
lbIdle: TListBox;
lblOnIdle: TLabel;
lbActionUpdate: TListBox;
lblOnActionUpdate: TLabel;
procedure ApplicationEventsActionExecute(Action: TBasicAction;
var Handled: Boolean);
procedure ApplicationEventsActionUpdate(Action: TBasicAction;
var Handled: Boolean);
procedure ApplicationEventsActivate(Sender: TObject);
procedure ApplicationEventsDeactivate(Sender : TObject);
procedure ApplicationEventsException(Sender : TObject; E: Exception);
function ApplicationEventsHelp(Command: Word; Data: Integer;
var CallHelp: Boolean): Boolean;
procedure ApplicationEventsHint(Sender: TObject);
procedure ApplicationEventsIdle(Sender: TObject; var Done: Boolean);
procedure ApplicationEventsMessage(var Msg: tagMSG;

Font.Height = -16
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
end
<-------> object Label2: TLabel
Left = 32
Top = 56
Width = 174
Height = 13
Caption = 'Copyright 1997, Borland International'
end
<-------> object Button1: TButton
Left = 82
Top = 96
Width = 75
Height = 25
Caption = 'OK'
TabOrder = 0
OnClick = Button1Click
end
end

ObjectS
الملف لبرنامج
(AppEvents)
إعداد **علاء الدين اللباد**
هاتف ٠٩٤٤٥٧٥٣٧١
unit main;
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
AppEvnts, ActnList, Menus, StdCtrls;

```
OnActionUpdate');
end;

procedure TMainForm.ApplicationEventsActivate(Sender: TObject);
begin
  lbOther.Items.Add('Event OnActivate');
end;

procedure TMainForm.ApplicationEventsDeactivate(Sender: TObject);
begin
  lbOther.Items.Add('Event OnDeactivate');
end;

procedure TMainForm.ApplicationEventsException(Sender: TObject;
  E: Exception);
begin
  lbOther.Items.Add('Event OnException: ' + E.Message);
end;

function TMainForm.ApplicationEventsHelp(Command: Word; Data: Integer;
  var CallHelp: Boolean): Boolean;
begin
  lbOther.Items.Add('Event OnHelp');
  Result := False;
end;

procedure TMainForm.ApplicationEventsHint(Sender: TObject);
begin
  lbOther.Items.Add('Event OnHint');
end;

procedure TMainForm.ApplicationEventsIdle(Sender: TObject;
```

```
      var Handled: Boolean);
    procedure ApplicationEventsMinimize(Sender: TObject);
    procedure ApplicationEventsRestore(Sender: TObject);
    procedure ApplicationEventsShortCut(var Msg: TWMKey;
      var Handled: Boolean);
    procedure ApplicationEventsShowHint(var HintStr: String;
      var CanShow: Boolean; var HintInfo: THintInfo);
    procedure ActionExecute(Sender: TObject);
    procedure MenuExceptionClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  MainForm: TMainForm;

implementation

{$R *.dfm}

procedure TMainForm.ApplicationEventsActionExecute(Action: TBasicAction;
  var Handled: Boolean);
begin
  lbOther.Items.Add('Event OnActionExecute');
end;

procedure TMainForm.ApplicationEventsActionUpdate(Action: TBasicAction;
  var Handled: Boolean);
begin
  lbActionUpdate.Items.Add('Event
```

```
procedure TMainForm.ActionExecute(Sender: TObject);
begin
  ShowMessage('Action executed');
end;

procedure TMainForm.MenuExceptionClick(Sender: TObject);
resourcestring
  sExceptionRaised = 'This is an exception';

begin
  raise Exception.Create(sExceptionRaised);
end;

end.
```

الملف النصي للبرنامج

Scroll bar

وفيه

Objects 40

من إعداد

علاء الدين اللباد

٠٩٤٤٥٧٥٣٧١

```
<-------> object ScrollForm1: TScrollForm1
  Left = 181
  Top = 156
  BorderIcons = [biSystemMenu, biMinimize]
  BorderStyle = bsSingle
  Caption = 'ScrollDemo'
  ClientHeight = 194
  ClientWidth = 436
  Color = clBtnFace
```

```
  var Done: Boolean);
begin
  lbIdle.Items.Add('Event OnIdle');
end;

procedure TMainForm.ApplicationEventsMessage(var Msg: tagMSG;
  var Handled: Boolean);
begin
  lbOnMessage.Items.Add('X:=' + IntToStr(Msg.pt.x) + '|Y:=' + IntToStr(Msg.pt.y));
end;

procedure TMainForm.ApplicationEventsMinimize(Sender: TObject);
begin
  lbOther.Items.Add('Event OnMinimize');
end;

procedure TMainForm.ApplicationEventsRestore(Sender: TObject);
begin
  lbOther.Items.Add('Event OnRestore');
end;

procedure TMainForm.ApplicationEventsShortCut(var Msg: TWMKey;
  var Handled: Boolean);
begin
  lbOther.Items.Add('Event OnShortCut');
end;

procedure TMainForm.ApplicationEventsShowHint(var HintStr: String;
  var CanShow: Boolean; var HintInfo: THintInfo);
begin
  lbOther.Items.Add('Event OnShowHint');
end;
```

```
   Left = 376
   Top = 16
   Width = 35
   Height = 13
   Caption = 'Vertical'
  end
  <-------> object HScrollb:
 TScrollBar
   Left = 9
   Top = 169
   Width = 159
   Height = 16
   Max = 32767
   PageSize = 0
   TabOrder = 1
   OnScroll = HScrollbScroll
  end
  <-------> object VScrollb:
 TScrollBar
   Left = 169
   Top = 9
   Width = 16
   Height = 157
   Kind = sbVertical
   PageSize = 0
   TabOrder = 2
   OnChange = VScrollbChange
  end
  <-------> object HUnits: TEdit
   Left = 275
   Top = 38
   Width = 48
   Height = 21
   ReadOnly = True
   TabOrder = 3
   Text = '1'
  end
  <-------> object HPage: TEdit
   Left = 275
   Top = 74
   Width = 48
   Height = 21
   ReadOnly = True
   TabOrder = 4
   Text = '0'
  end
  <-------> object HMax: TEdit
   Left = 276
```

```
  Font.Charset =
 DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  Menu = MainMenu1
  OldCreateOrder = True
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Bevel1: TBevel
   Left = 0
   Top = 0
   Width = 436
   Height = 9
   Align = alTop
   Shape = bsTopLine
  end
  <-------> object Label3: TLabel
   Left = 238
   Top = 42
   Width = 24
   Height = 13
   Caption = 'Units'
  end
  <-------> object Label4: TLabel
   Left = 198
   Top = 78
   Width = 64
   Height = 13
   Caption = 'LargeChange'
  end
  <-------> object Label5: TLabel
   Left = 242
   Top = 117
   Width = 20
   Height = 13
   Caption = 'Max'
  end
  <-------> object Label1: TLabel
   Left = 280
   Top = 16
   Width = 47
   Height = 13
   Caption = 'Horizontal'
  end
  <-------> object Vertical: TLabel
```

```
TUpDown
   Left = 323
   Top = 74
   Width = 15
   Height = 21
   Associate = HPage
   Max = 32767
   TabOrder = 10
   OnClick = UpDownClick
  end
  <-------> object HMaxUpDown: TUpDown
   Left = 324
   Top = 113
   Width = 15
   Height = 21
   Associate = HMax
   Max = 32767
   TabOrder = 11
   OnClick = UpDownClick
  end
  <-------> object VUnitsUpDown: TUpDown
   Left = 411
   Top = 38
   Width = 15
   Height = 21
   Associate = VUnits
   Min = 1
   Max = 32767
   Position = 1
   TabOrder = 12
   OnClick = UpDownClick
  end
  <-------> object VMaxUpDown: TUpDown
   Left = 411
   Top = 113
   Width = 15
   Height = 21
   Associate = VMax
   Max = 32767
   TabOrder = 13
   OnClick = UpDownClick
  end
  <-------> object VPageUpDown: TUpDown
   Left = 411
   Top = 113
   Width = 48
   Height = 21
   ReadOnly = True
   TabOrder = 5
   Text = '0'
  end
  <-------> object VUnits: TEdit
   Left = 363
   Top = 38
   Width = 48
   Height = 21
   ReadOnly = True
   TabOrder = 6
   Text = '1'
  end
  <-------> object VPage: TEdit
   Left = 363
   Top = 74
   Width = 48
   Height = 21
   ReadOnly = True
   TabOrder = 7
   Text = '0'
  end
  <-------> object VMax: TEdit
   Left = 363
   Top = 113
   Width = 48
   Height = 21
   ReadOnly = True
   TabOrder = 8
   Text = '0'
  end
  <-------> object HUnitsUpDown: TUpDown
   Left = 323
   Top = 38
   Width = 15
   Height = 21
   Associate = HUnits
   Min = 1
   Max = 32767
   Position = 1
   TabOrder = 9
   OnClick = UpDownClick
  end
  <-------> object HPageUpDown:
```

```
  <-------> object MainMenu1: TMainMenu
    Left = 196
    Top = 159
    <-------> object File1: TMenuItem
      Caption = '&File'
      <-------> object Open1: TMenuItem
        Caption = '&Open...'
        OnClick = Open1Click
      end
      <-------> object N1: TMenuItem
        Caption = '-'
      end
      <-------> object Exit1: TMenuItem
        Caption = 'E&xit'
        OnClick = Exit1Click
      end
    end
    <-------> object View1: TMenuItem
      Caption = '&Scroll'
      OnClick = View1Click
      <-------> object VScrollMenu: TMenuItem
        Caption = '&Vertical'
        OnClick = VScrollMenuClick
      end
      <-------> object HScrollMenu: TMenuItem
        Caption = '&Horizontal'
        OnClick = HScrollMenuClick
      end
    end
    <-------> object Help1: TMenuItem
      Caption = '&Help'
      <-------> object About1: TMenuItem
        Caption = '&About...'
        OnClick = About1Click
      end
    end
  end
  <-------> object OpenDialog: TOpenDialog
    DefaultExt = '.bmp'
```

```
    Top = 74
    Width = 15
    Height = 21
    Associate = VPage
    Max = 32767
    TabOrder = 14
    OnClick = UpDownClick
  end
  <-------> object DefaultBtn: TButton
    Left = 348
    Top = 157
    Width = 75
    Height = 25
    Caption = '&Defaults'
    ParentShowHint = False
    ShowHint = False
    TabOrder = 15
    OnClick = DefaultBtnClick
  end
  <-------> object ApplyBtn: TButton
    Left = 265
    Top = 157
    Width = 75
    Height = 25
    Caption = '&Apply'
    Enabled = False
    TabOrder = 16
    OnClick = ApplyBtnClick
  end
  <-------> object ImgPanel: TPanel
    Left = 8
    Top = 8
    Width = 160
    Height = 160
    BevelOuter = bvNone
    BorderStyle = bsSingle
    TabOrder = 0
    <-------> object Image: TImage
      Left = 0
      Top = 0
      Width = 600
      Height = 470
      AutoSize = True
      Picture.Data = {
        }
    end
  end
```

```
  Filter =
    'Images
(*.bmp;*.ico)|*.bmp;*.ico|Bitmaps
(*.bmp)|*.bmp|Icons (*.' +
    'ico)|*.ico'
  Options = [ofHideReadOnly,
ofPathMustExist, ofFileMustExist]
  Left = 230
  Top = 158
 end
End

<-------> object AboutBox:
TAboutBox
 Left = 200
 Top = 108
 BorderIcons = [biSystemMenu]
 BorderStyle = bsDialog
 Caption = 'About ScrollDemo'
 ClientHeight = 135
 ClientWidth = 239
 Color = clBtnFace
 Font.Charset =
DEFAULT_CHARSET
 Font.Color = clWindowText
 Font.Height = -11
 Font.Name = 'MS Sans Serif'
 Font.Style = []
 OldCreateOrder = True
 Position = poScreenCenter
 PixelsPerInch = 96
 TextHeight = 13
 <-------> object Label1: TLabel
  Left = 24
  Top = 24
  Width = 191
  Height = 20
  Caption = 'ScrollBar
Demonstation'
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -16
  Font.Name = 'MS Sans Serif'
  Font.Style = [fsBold]
  ParentFont = False
 end
 <-------> object Label2: TLabel
  Left = 32
  Top = 56
  Width = 174
  Height = 13
  Caption = 'Copyright 1997,
Borland International'
 end
 <-------> object Button1: TButton
  Left = 82
  Top = 96
  Width = 75
  Height = 25
  Caption = 'OK'
  TabOrder = 0
  OnClick = Button1Click
 end
end
```

Objects$

الملف التنفيذي لبرنامج

(ConvertI)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit ConvertItUnit;

interface

uses
  Windows, Messages, SysUtils,
Variants, Classes, Graphics,
Controls, Forms,
  Dialogs, StdCtrls, ComCtrls;

type
  TForm1 = class(TForm)
    ConvTypes: TListBox;
    ConvValue: TEdit;
    ConvResults: TListBox;
    ConvValueIncDec: TUpDown;
    ConvFamilies: TTabControl;
    StatusBar1: TStatusBar;
    procedure FormShow(Sender:
TObject);
    procedure
ConvTypesClick(Sender: TObject);
    procedure
ConvValueChange(Sender:
TObject);
    procedure
ConvFamiliesChange(Sender:
TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

uses
  ConvUtils, StdConvs, EuroConv,
StrUtils;

procedure
TForm1.FormShow(Sender:
TObject);
var
  LFamilies: TConvFamilyArray;
  I: Integer;
  LStrings: TStringList;
begin
  ConvFamilies.Tabs.Clear;
  LStrings := TStringList.Create;
  try
    GetConvFamilies(LFamilies);
    for I := 0 to Length(LFamilies) - 1
do

LStrings.AddObject(ConvFamilyTo
Description(LFamilies[I]),
TObject(LFamilies[I]));
    LStrings.Sort;

ConvFamilies.Tabs.Assign(LStrings)
;
    ConvFamiliesChange(Sender);
  finally
    LStrings.Free;
  end;
end;

procedure
TForm1.ConvFamiliesChange(Send
er: TObject);
var
  LFamily: TConvFamily;
  LTypes: TConvTypeArray;
  I: Integer;
begin
  LFamily :=
TConvFamily(ConvFamilies.Tabs.O
bjects[ConvFamilies.TabIndex]);
  with ConvTypes, Items do
  begin
    BeginUpdate;
```

```
    Clear;
    GetConvTypes(LFamily, LTypes);
    for I := 0 to Length(LTypes) - 1 do

AddObject(ConvTypeToDescription
(LTypes[I]), TObject(LTypes[I]));
    ItemIndex := 0;
    EndUpdate;
  end;
  ConvTypesClick(Sender);
end;

procedure
TForm1.ConvTypesClick(Sender:
TObject);
begin
  ConvValueChange(Sender);
end;

procedure
TForm1.ConvValueChange(Sender:
TObject);
var
  LValue: Double;
  LBaseType, LTestType:
TConvType;
  I: Integer;
begin
  with ConvResults, Items do
  try
    BeginUpdate;
    Clear;
    try
      LValue :=
StrToFloatDef(ConvValue.Text, 0);
      if ConvTypes.ItemIndex <> -1
then
      begin
        LBaseType :=
TConvType(ConvTypes.Items.Objec
ts[ConvTypes.ItemIndex]);
        for I := 0 to
ConvTypes.Items.Count - 1 do
        begin
          LTestType :=
TConvType(ConvTypes.Items.Objec
ts[I]);
          Add(Format('%n %s',
[Convert(LValue, LBaseType,
LTestType),

ConvTypeToDescription(LTestType
)]));
        end;
      end
      else
        Add('No base type');
    except
      Add('Cannot parse value');
    end;
  finally
    EndUpdate;
  end;
end;

end.
```

ObjectS

الملف النصي لبرنامج

(ConvertI)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
object Form1: TForm1
  Left = 192
  Top = 107
  Width = 420
  Height = 466
  Caption = 'Conversion Tester'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  OnShow = FormShow
  PixelsPerInch = 96
  TextHeight = 13
  object ConvFamilies: TTabControl
```

```
    Left = 0
    Top = 0
    Width = 412
    Height = 413
    Align = alClient
    TabOrder = 0
    Tabs.Strings = (
      'Testing')
    TabIndex = 0
    OnChange =
ConvFamiliesChange
    DesignSize = (
      412
      413)
    object ConvTypes: TListBox
      Left = 16
      Top = 32
      Width = 177
      Height = 372
      Anchors = [akLeft, akTop,
akBottom]
      ItemHeight = 13
      Sorted = True
      TabOrder = 0
      OnClick = ConvTypesClick
    end
    object ConvValueIncDec:
TUpDown
      Left = 377
      Top = 32
      Width = 16
      Height = 21
      Anchors = [akTop, akRight]
      Associate = ConvValue
      Min = -32000
      Max = 32000
      Position = 1
      TabOrder = 2
      Thousands = False
    end
    object ConvResults: TListBox
      Left = 200
      Top = 56
      Width = 196
      Height = 348
      TabStop = False
      Anchors = [akLeft, akTop,
akRight, akBottom]
      ItemHeight = 13
      ParentColor = True
      TabOrder = 3
    end
    object ConvValue: TEdit
      Left = 200
      Top = 32
      Width = 177
      Height = 21
      Anchors = [akLeft, akTop,
akRight]
      TabOrder = 1
      Text = '1'
      OnChange = ConvValueChange
    end
  end
  object StatusBar1: TStatusBar
    Left = 0
    Top = 413
    Width = 412
    Height = 19
    Panels = <
      item
        Width = 50
      end>
  end
End
```

الملف التنفيذي لبرنامج

(Prgrsba)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit Pg1;

Interface

Uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ComCtrls, ExtCtrls, Pg2;

Const
  WM_ThreadDoneMsg = WM_User + 8;

Type
  TForm1 = class(TForm)
    ProgressBar1: TProgressBar;
    ProgressBar2: TProgressBar;
    Button1: TButton;
    Button2: TButton;
    TrackBar1: TTrackBar;
    TrackBar2: TTrackBar;
    Bevel1: TBevel;
    Bevel2: TBevel;
    Label1: TLabel;
    Label2: TLabel;
    Button3: TButton;
    Button4: TButton;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure TrackBar1Change(Sender: TObject);
    procedure TrackBar2Change(Sender: TObject);
    procedure FormDestroy(Sender: TObject);
  private
    { Private declarations }
    MyThread1 : TMyThread; // thread number 1
    MyThread2 : TMyThread; // thread number 2
    Thread1Active : boolean; // used to test if thread 1 is active
    Thread2Active : boolean; // used to test if thread 2 is active
    procedure ThreadDone(var AMessage : TMessage); message WM_ThreadDoneMsg; // Message to be sent back from thread when its done
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

procedure TForm1.Button1Click(Sender: TObject); // Create Thread 1
{ The thread will destroy iteself when it is done executing because FreeOnTerminate is set to true.
The first paramter is the priority, and the second is the progressbar to update.
}
begin
   if (MyThread1 = nil) or (Thread1Active = false) then // make sure its not already running
   begin
```

```
  Else
   ShowMessage('Thread not started');
end;

procedure TForm1.ThreadDone(var AMessage: TMessage); // keep track of when and which thread is done executing
begin
  if ((MyThread1 <> nil) and (MyThread1.ThreadID = cardinal(AMessage.WParam))) then
  begin
     Thread1Active := false;
  end;
  if ((MyThread2 <> nil) and (MyThread2.ThreadID = cardinal(AMessage.WParam))) then
  begin
     Thread2Active := false;
  end;
end;


procedure TForm1.FormCreate(Sender: TObject); // initialize to zero
begin
  Thread1Active := false;
  Thread2Active := false;
end;


procedure TForm1.TrackBar1Change(Sender: TObject); // set Thread 1 Priority
begin
  if (MyThread1 <> nil) and (Thread1Active = true) then
     MyThread1.priority := TThreadPriority(TrackBar1.Position);
end;

procedure TForm1.TrackBar2Change(Sender: TObject); // set Thread 2 Priority
begin
    MyThread1 := TMyThread.CreateIt(TrackBar1.Position, ProgressBar1);
    Thread1Active := true;
  End
  Else
    ShowMessage('Thread still executing');
end;

procedure TForm1.Button2Click(Sender: TObject); // Create Thread 2
Begin
  if (MyThread2 = nil) or (Thread2Active = false) then  // make sure its not already running
  Begin
    MyThread2 := TMyThread.CreateIt(TrackBar2.Position, ProgressBar2);
    Thread2Active := true;
  End
  Else
    ShowMessage('Thread still executing');
end;

procedure TForm1.Button3Click(Sender: TObject); // Terminate Thread 1
Begin
 if (MyThread1 <> nil) and (Thread1Active = true) then  // check to see if it is running
   MyThread1.Terminate
 Else
  ShowMessage('Thread not started');
end;

procedure TForm1.Button4Click(Sender: TObject); // Terminate Thread 2
Begin
 if (MyThread2 <> nil) and (Thread2Active = true) then  // check to see if it is running
   MyThread2.Terminate
```

```
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  Position = poDesktopCenter
  OnCreate = FormCreate
  OnDestroy = FormDestroy
  PixelsPerInch = 96
  TextHeight = 13
  object Bevel1: TBevel
    Left = 32
    Top = 8
    Width = 513
    Height = 169
  end
  object Bevel2: TBevel
    Left = 32
    Top = 200
    Width = 513
    Height = 169
  end
  object Label1: TLabel
    Left = 256
    Top = 104
    Width = 34
    Height = 13
    Caption = 'Priority:'
  end
  object Label2: TLabel
    Left = 248
    Top = 296
    Width = 34
    Height = 13
    Caption = 'Priority:'
  end
  object ProgressBar1: TProgressBar
    Left = 64
    Top = 32
    Width = 393
    Height = 25
    Step = 1
    TabOrder = 0
  end
  object ProgressBar2: TProgressBar
```

```
  if (MyThread2 <> nil) and (Thread2Active = true) then
    MyThread2.priority := TThreadPriority(TrackBar2.Position);
end;


procedure TForm1.FormDestroy(Sender: TObject); // Terminate any threads still running
Begin
   if (MyThread1 <> nil) and (Thread1Active = true) then
   Begin
     MyThread1.Terminate;
     MyThread1.WaitFor;  // wait for it to terminate
   end;
   if (MyThread2 <> nil) and (Thread2Active = true) then
   Begin
     MyThread2.Terminate;
     MyThread2.WaitFor;
   end;
end;

end.
```

ObjectS
الملف النصي لبرنامج
(Prgrsba)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
object Form1: TForm1
  Left = 169
  Top = 109
  Width = 712
  Height = 507
  Caption = 'TThread, TTrackBar, TProgressBar'
```

```
    Left = 64
    Top = 224
    Width = 401
    Height = 25
    TabOrder = 1
  end
  object Button1: TButton
    Left = 72
    Top = 80
    Width = 105
    Height = 25
    Caption = 'Create Thread'
    TabOrder = 2
    OnClick = Button1Click
  end
  object Button2: TButton
    Left = 72
    Top = 272
    Width = 105
    Height = 25
    Caption = 'Create Thread'
    TabOrder = 3
    OnClick = Button2Click
  end
  object TrackBar1: TTrackBar
    Left = 296
    Top = 104
    Width = 193
    Height = 33
    Max = 6
    Position = 3
    TabOrder = 4
    OnChange = TrackBar1Change
  end
  object TrackBar2: TTrackBar
    Left = 288
    Top = 296
    Width = 193
    Height = 45
    Max = 6
    Position = 3
    TabOrder = 5
    OnChange = TrackBar2Change
  end
  object Button3: TButton
    Left = 72
    Top = 128
    Width = 105
    Height = 25
    Caption = 'Terminate Thread'
    TabOrder = 6
    OnClick = Button3Click
  end
  object Button4: TButton
    Left = 72
    Top = 320
    Width = 105
    Height = 25
    Caption = 'Terminate Thread'
    TabOrder = 7
    OnClick = Button4Click
  end
End
```

```
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure LCDNumber1MouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
  private
    { Private declarations }
    FRunning: Boolean;
    procedure SetRunning(Value: Boolean);
  public
    { Public declarations }
    StartTime: Extended;
    ElapsedTime: Extended;
    Reset: Boolean;
    property Running: Boolean read FRunning write SetRunning;
  end;

var
  Form1: TForm1;

implementation

{$R *.xfm}

procedure TForm1.SetRunning(Value: Boolean);
begin
  if FRunning <> Value then
  begin
    FRunning := Value;
    StartTime := ElapsedTime;
    if Value then Reset := False;
  end;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
  if Reset then ElapsedTime := Now
```

الملف التنفيذي لبرنامج

(----StopWatch)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
program stopwatch;

uses
  QForms,
  main in 'main.pas' {Form1};

{$R *.res}

begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.Run;
end.

\\
unit main;

interface

uses
  SysUtils, Types, Classes, QGraphics, QControls, QForms, QDialogs,
  QStdCtrls, QTypes, QExtCtrls;

type
  TForm1 = class(TForm)
    LCDNumber1: TLCDNumber;
    Timer1: TTimer;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    procedure Timer1Timer(Sender: TObject);
```

```
      if NewBorderStyle >
Ord(High(TBorderStyle)) then
        NewBorderStyle :=
Ord(Low(TBorderStyle));
      LCDNumber1.BorderStyle :=
TBorderStyle(NewBorderStyle);
    end;
   mbRight:
    begin
      NewSegmentStyle :=
Ord(LCDNumber1.SegmentStyle) +
1;
      if NewSegmentStyle >
Ord(High(TLCDSegmentStyle))
then
        NewSegmentStyle :=
Ord(Low(TLCDSegmentStyle));
      LCDNumber1.SegmentStyle :=
TLCDSegmentStyle(NewSegmentSt
yle);
    end;
  end;
end;

end.
```

ObjectS
الملف لبرنامج
(----StopWatch)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
object Form1: TForm1
  Left = 117
  Top = 167
  Width = 237
  Height = 67
  VertScrollBar.Range = 65
  HorzScrollBar.Range = 233
  ActiveControl = Button1
  Caption = 'Form1'
  Color = clBackground
  PixelsPerInch = 75
  TextHeight = 13
```

```
else ElapsedTime := Now -
StartTime;
  if Running then
    LCDNumber1.Value :=
FormatDateTime('nn:ss.zzz',
ElapsedTime);
end;

procedure
TForm1.Button1Click(Sender:
TObject);
begin
  Running := True;
end;

procedure
TForm1.Button2Click(Sender:
TObject);
begin
  Running := False;
end;

procedure
TForm1.Button3Click(Sender:
TObject);
begin
  if not Running then
  begin
    LCDNumber1.Value :=
'00:00.000';
    Reset := True;
  end;
end;

procedure
TForm1.LCDNumber1MouseUp(Se
nder: TObject; Button:
TMouseButton;
  Shift: TShiftState; X, Y: Integer);
var
  NewBorderStyle: Integer;
  NewSegmentStyle: Integer;
begin
  case Button of
    mbLeft:
      begin
        NewBorderStyle :=
Ord(LCDNumber1.BorderStyle) +
1;
```

```
  TextWidth = 6
  object LCDNumber1: TLCDNumber
    Left = 0
    Top = 0
    Width = 237
    Height = 35
    Align = alTop
    Color = clBackground
    Digits = 9
    ParentColor = False
    SegmentStyle = ssFilled
    Value = '00:00.000'
    OnMouseUp = LCDNumber1MouseUp
  end
  object Button1: TButton
    Left = 2
    Top = 40
    Width = 75
    Height = 25
    Caption = 'Start'
    TabOrder = 1
    OnClick = Button1Click
  end
  object Button2: TButton
    Left = 80
    Top = 40
    Width = 75
    Height = 25
    Caption = 'Stop'
    TabOrder = 2
    OnClick = Button2Click
  end
  object Button3: TButton
    Left = 158
    Top = 40
    Width = 75
    Height = 25
    Caption = 'Reset'
    TabOrder = 3
    OnClick = Button3Click
  end
  object Timer1: TTimer
    Interval = 1
    OnTimer = Timer1Timer
    Left = 8
    Top = 4
  end
End
```

الملف التنفيذي لبرنامج

(ImagView)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit ImageWin;

interface

uses Windows, Classes, Graphics, Forms, Controls,
  FileCtrl, StdCtrls, ExtCtrls, Buttons, Spin, ComCtrls, Dialogs;

type
  TImageForm = class(TForm)
    DirectoryListBox1: TDirectoryListBox;
    DriveComboBox1: TDriveComboBox;
    FileEdit: TEdit;
    UpDownGroup: TGroupBox;
    SpeedButton1: TSpeedButton;
    BitBtn1: TBitBtn;
    DisabledGrp: TGroupBox;
    SpeedButton2: TSpeedButton;
    BitBtn2: TBitBtn;
    Panel1: TPanel;
    Image1: TImage;
    FileListBox1: TFileListBox;
    Label2: TLabel;
    ViewBtn: TBitBtn;
    Bevel1: TBevel;
    Bevel2: TBevel;
    FilterComboBox1: TFilterComboBox;
    GlyphCheck: TCheckBox;
    StretchCheck: TCheckBox;
    UpDownEdit: TEdit;
    UpDown1: TUpDown;
    procedure FileListBox1Click(Sender: TObject);
    procedure ViewBtnClick(Sender: TObject);
    procedure ViewAsGlyph(const FileExt: string);
    procedure GlyphCheckClick(Sender: TObject);
    procedure StretchCheckClick(Sender: TObject);
    procedure FileEditKeyPress(Sender: TObject; var Key: Char);
    procedure UpDownEditChange(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    FormCaption: string;
  end;

var
  ImageForm: TImageForm;

implementation

uses ViewWin, SysUtils;

{$R *.dfm}

procedure TImageForm.FileListBox1Click(Sender: TObject);
var
  FileExt: string[4];
begin
  FileExt := AnsiUpperCase(ExtractFileExt(FileListBox1.Filename));
  if (FileExt = '.BMP') or (FileExt = '.ICO') or (FileExt = '.WMF') or
    (FileExt = '.EMF') then
  begin
    Image1.Picture.LoadFromFile(FileL
```

```
Image1.Picture.Bitmap;
   SpeedButton2.Glyph := Image1.Picture.Bitmap;
   UpDown1.Position := SpeedButton1.NumGlyphs;
   BitBtn1.Glyph := Image1.Picture.Bitmap;
   BitBtn2.Glyph := Image1.Picture.Bitmap;
  end
  else begin
   SpeedButton1.Glyph := nil;
   SpeedButton2.Glyph := nil;
   BitBtn1.Glyph := nil;
   BitBtn2.Glyph := nil;
  end;
  UpDown1.Enabled := GlyphCheck.Checked;
  UpDownEdit.Enabled := GlyphCheck.Checked;
  Label2.Enabled := GlyphCheck.Checked;
end;

procedure TImageForm.ViewBtnClick(Sender: TObject);
begin
  ViewForm.HorzScrollBar.Range := Image1.Picture.Width;
  ViewForm.VertScrollBar.Range := Image1.Picture.Height;
  ViewForm.Caption := Caption;
  ViewForm.Show;
  ViewForm.WindowState := wsNormal;
end;

procedure TImageForm.UpDownEditChange(Sender: TObject);
resourcestring
  sMinValue = 'Value below minimum';
  sMaxValue = 'Value over maximum';

begin
  if (StrToInt(UpDownEdit.Text) <
```

```
istBox1.Filename);
   Caption := FormCaption + ExtractFilename(FileListBox1.Filename);
   if (FileExt = '.BMP') then
   begin
    Caption := Caption +
      Format(' (%d x %d)', [Image1.Picture.Width, Image1.Picture.Height]);
    ViewForm.Image1.Picture := Image1.Picture;
    ViewForm.Caption := Caption;
    if GlyphCheck.Checked then ViewAsGlyph(FileExt);
   end
   else
    GlyphCheck.Checked := False;
   if FileExt = '.ICO' then
   begin
    Icon := Image1.Picture.Icon;
    ViewForm.Image1.Picture.Icon := Icon;
   end;
   if (FileExt = '.WMF') or (FileExt = '.EMF') then
     ViewForm.Image1.Picture.Metafile := Image1.Picture.Metafile;
  end;
end;

procedure TImageForm.GlyphCheckClick(Sender: TObject);
begin
  ViewAsGlyph(AnsiUpperCase(ExtractFileExt(FileListBox1.Filename)));
end;

procedure TImageForm.ViewAsGlyph(const FileExt: string);
begin
  if GlyphCheck.Checked and (FileExt = '.BMP') then
  begin
   SpeedButton1.Glyph :=
```

```
end.

\\\\\\\\\\\\\\\\\\\\
```

ObjectS

الملف لبرنامج

(ImagView)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
object ImageForm: TImageForm
  Left = 193
  Top = 107
  ActiveControl = FileEdit
  BorderIcons = [biSystemMenu, biMinimize]
  BorderStyle = bsSingle
  Caption = 'Image Viewer'
  ClientHeight = 315
  ClientWidth = 502
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poDefaultPosOnly
  PixelsPerInch = 96
  TextHeight = 13
  object Bevel1: TBevel
    Left = 316
    Top = 5
    Width = 171
    Height = 121
  end
  object Bevel2: TBevel
    Left = 317
    Top = 132
    Width = 170
    Height = 177
  end
  object Label1: TLabel
    Left = 325
    Top = 11
    Width = 147
    Height = 13
```

```
UpDown1.Min) then
  begin
    UpDownEdit.Text := '1';
    raise ERangeError.Create(sMinValue);
  end
  else if (StrToInt(UpDownEdit.Text) > UpDown1.Max) then
  begin
    UpDownEdit.Text := '4';
    raise ERangeError.Create(sMaxValue);
  end;

  SpeedButton1.NumGlyphs := UpDown1.Position;
  SpeedButton2.NumGlyphs := UpDown1.Position;
end;

procedure TImageForm.StretchCheckClick(Sender: TObject);
begin
  Image1.Stretch := StretchCheck.Checked;
end;

procedure TImageForm.FileEditKeyPress(Sender: TObject; var Key: Char);
begin
  if Key = #13 then
  begin
    FileListBox1.ApplyFilePath(FileEdit.Text);
    Key := #0;
  end;
end;

procedure TImageForm.FormCreate(Sender: TObject);
begin
  FormCaption := Caption + ' - ';
  UpDown1.Min := 1;
  UpDown1.Max := 4;
end;
```

```
    Left = 116
    Top = 26
    Width = 25
    Height = 25
    AllowAllUp = True
    GroupIndex = 1
  end
  object BitBtn1: TBitBtn
    Left = 12
    Top = 18
    Width = 92
    Height = 33
    Caption = 'BitBtn1'
    TabOrder = 0
  end
 end
 object DisabledGrp: TGroupBox
  Left = 323
  Top = 220
  Width = 154
  Height = 60
  Caption = 'Disabled'
  TabOrder = 7
  object SpeedButton2:
TSpeedButton
    Left = 116
    Top = 25
    Width = 25
    Height = 25
    Enabled = False
  end
  object BitBtn2: TBitBtn
    Left = 11
    Top = 18
    Width = 92
    Height = 33
    Caption = 'BitBtn2'
    Enabled = False
    TabOrder = 0
  end
 end
 object Panel1: TPanel
  Left = 324
  Top = 27
  Width = 153
  Height = 68
  BevelInner = bvLowered
  TabOrder = 8
    AutoSize = False
    Caption = 'Preview'
  end
  object Label2: TLabel
    Left = 324
    Top = 290
    Width = 93
    Height = 13
    Caption = 'Number of Glyphs - '
  end
  object DirectoryListBox1:
TDirectoryListBox
    Left = 8
    Top = 12
    Width = 148
    Height = 260
    FileList = FileListBox1
    IntegralHeight = True
    ItemHeight = 16
    TabOrder = 1
  end
  object DriveComboBox1:
TDriveComboBox
    Left = 9
    Top = 277
    Width = 148
    Height = 19
    DirList = DirectoryListBox1
    TabOrder = 3
  end
  object FileEdit: TEdit
    Left = 166
    Top = 13
    Width = 139
    Height = 21
    TabOrder = 0
    Text = '*.bmp;*.ico;*.wmf;*.emf'
    OnKeyPress = FileEditKeyPress
  end
  object UpDownGroup: TGroupBox
    Left = 323
    Top = 158
    Width = 154
    Height = 60
    Caption = 'Up / Down'
    TabOrder = 5
    object SpeedButton1:
TSpeedButton
```

```
    Top = 137
    Width = 104
    Height = 17
    Caption = 'View as Glyph'
    TabOrder = 9
    OnClick = GlyphCheckClick
  end
  object StretchCheck: TCheckBox
    Left = 413
    Top = 96
    Width = 64
    Height = 17
    Caption = 'Stretch'
    TabOrder = 11
    OnClick = StretchCheckClick
  end
  object UpDownEdit: TEdit
    Left = 439
    Top = 286
    Width = 24
    Height = 21
    TabOrder = 10
    Text = '1'
    OnChange = UpDownEditChange
  end
  object UpDown1: TUpDown
    Left = 463
    Top = 286
    Width = 11
    Height = 21
    Associate = UpDownEdit
    Position = 1
    TabOrder = 12
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
object ViewForm: TViewForm
  Left = 187
  Top = 458
  AutoScroll = False
  Caption = 'View Form'
  ClientHeight = 287
  ClientWidth = 495
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
```

```
    object Image1: TImage
      Left = 2
      Top = 2
      Width = 149
      Height = 64
      Align = alClient
    end
  end
  object FileListBox1: TFileListBox
    Left = 166
    Top = 41
    Width = 139
    Height = 227
    FileEdit = FileEdit
    ItemHeight = 13
    Mask = '*.bmp;*.ico;*.wmf;*.emf'
    TabOrder = 2
    OnClick = FileListBox1Click
  end
  object ViewBtn: TBitBtn
    Left = 325
    Top = 98
    Width = 63
    Height = 24
    Caption = '&Full View'
    TabOrder = 6
    OnClick = ViewBtnClick
  end
  object FilterComboBox1:
TFilterComboBox
    Left = 166
    Top = 275
    Width = 140
    Height = 21
    FileList = FileListBox1
    Filter =
      'Image Files (*.bmp, *.ico,
*.wmf,
*.emf)|*.bmp;*.ico;*.wmf;*.emf' +
      '|Bitmap Files
(*.bmp)|*.bmp|Icons
(*.ico)|*.ico|Metafiles (*.wmf' +
      ', *.emf)|*.wmf;*.emf|All files
(*.*)|*.*'
    TabOrder = 4
  end
  object GlyphCheck: TCheckBox
    Left = 327
```

```
  Left = 325
  Top = 11
  Width = 147
  Height = 13
  AutoSize = False
  Caption = 'Preview'
 end
 object Label2: TLabel
  Left = 324
  Top = 290
  Width = 93
  Height = 13
  Caption = 'Number of Glyphs - '
 end
 object DirectoryListBox1: TDirectoryListBox
  Left = 8
  Top = 12
  Width = 148
  Height = 260
  FileList = FileListBox1
  IntegralHeight = True
  ItemHeight = 16
  TabOrder = 1
 end
 object DriveComboBox1: TDriveComboBox
  Left = 9
  Top = 277
  Width = 148
  Height = 19
  DirList = DirectoryListBox1
  TabOrder = 3
 end
 object FileEdit: TEdit
  Left = 166
  Top = 13
  Width = 139
  Height = 21
  TabOrder = 0
  Text = '*.bmp;*.ico;*.wmf;*.emf'
  OnKeyPress = FileEditKeyPress
 end
 object UpDownGroup: TGroupBox
  Left = 323
  Top = 158
  Width = 154
  Height = 60
```

```
 Font.Height = -13
 Font.Name = 'System'
 Font.Style = []
 OldCreateOrder = True
 Position = poDefault
 PixelsPerInch = 96
 TextHeight = 16
 object Image1: TImage
  Left = 0
  Top = 0
  Width = 495
  Height = 287
  Align = alClient
  AutoSize = True
 end
End

\\\\\\\\\\\\\\

object ImageForm: TImageForm
 Left = 193
 Top = 107
 ActiveControl = FileEdit
 BorderIcons = [biSystemMenu, biMinimize]
 BorderStyle = bsSingle
 Caption = 'Image Viewer'
 ClientHeight = 315
 ClientWidth = 502
 Color = clBtnFace
 ParentFont = True
 OldCreateOrder = True
 Position = poDefaultPosOnly
 PixelsPerInch = 96
 TextHeight = 13
 object Bevel1: TBevel
  Left = 316
  Top = 5
  Width = 171
  Height = 121
 end
 object Bevel2: TBevel
  Left = 317
  Top = 132
  Width = 170
  Height = 177
 end
 object Label1: TLabel
```

```
  Width = 153
  Height = 68
  BevelInner = bvLowered
  TabOrder = 8
  object Image1: TImage
    Left = 2
    Top = 2
    Width = 149
    Height = 64
    Align = alClient
  end
 end
 object FileListBox1: TFileListBox
  Left = 166
  Top = 41
  Width = 139
  Height = 227
  FileEdit = FileEdit
  ItemHeight = 13
  Mask = '*.bmp;*.ico;*.wmf;*.emf'
  TabOrder = 2
  OnClick = FileListBox1Click
 end
 object ViewBtn: TBitBtn
  Left = 325
  Top = 98
  Width = 63
  Height = 24
  Caption = '&Full View'
  TabOrder = 6
  OnClick = ViewBtnClick
 end
 object FilterComboBox1:
TFilterComboBox
  Left = 166
  Top = 275
  Width = 140
  Height = 21
  FileList = FileListBox1
  Filter =
    'Image Files (*.bmp, *.ico,
*.wmf,
*.emf)|*.bmp;*.ico;*.wmf;*.emf' +
    '|Bitmap Files
(*.bmp)|*.bmp|Icons
(*.ico)|*.ico|Metafiles (*.wmf' +
    ', *.emf)|*.wmf;*.emf|All files
(*.*)|*.*'
```

```
  Caption = 'Up / Down'
  TabOrder = 5
  object SpeedButton1:
TSpeedButton
    Left = 116
    Top = 26
    Width = 25
    Height = 25
    AllowAllUp = True
    GroupIndex = 1
  end
  object BitBtn1: TBitBtn
    Left = 12
    Top = 18
    Width = 92
    Height = 33
    Caption = 'BitBtn1'
    TabOrder = 0
  end
 end
 object DisabledGrp: TGroupBox
  Left = 323
  Top = 220
  Width = 154
  Height = 60
  Caption = 'Disabled'
  TabOrder = 7
  object SpeedButton2:
TSpeedButton
    Left = 116
    Top = 25
    Width = 25
    Height = 25
    Enabled = False
  end
  object BitBtn2: TBitBtn
    Left = 11
    Top = 18
    Width = 92
    Height = 33
    Caption = 'BitBtn2'
    Enabled = False
    TabOrder = 0
  end
 end
 object Panel1: TPanel
  Left = 324
  Top = 27
```

```
    TabOrder = 4
  end
  object GlyphCheck: TCheckBox
    Left = 327
    Top = 137
    Width = 104
    Height = 17
    Caption = 'View as Glyph'
    TabOrder = 9
    OnClick = GlyphCheckClick
  end
  object StretchCheck: TCheckBox
    Left = 413
    Top = 96
    Width = 64
    Height = 17
    Caption = 'Stretch'
    TabOrder = 11
    OnClick = StretchCheckClick
  end
  object UpDownEdit: TEdit
    Left = 439
    Top = 286
    Width = 24
    Height = 21
    TabOrder = 10
    Text = '1'
    OnChange = UpDownEditChange
  end
  object UpDown1: TUpDown
    Left = 463
    Top = 286
    Width = 11
    Height = 21
    Associate = UpDownEdit
    Position = 1
    TabOrder = 12
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

الملف لبرنامج

(Threads)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit ThSort;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  ExtCtrls, StdCtrls;

type
  TThreadSortForm = class(TForm)
    StartBtn: TButton;
    BubbleSortBox: TPaintBox;
    SelectionSortBox: TPaintBox;
    QuickSortBox: TPaintBox;
    Label1: TLabel;
    Bevel1: TBevel;
    Bevel2: TBevel;
    Bevel3: TBevel;
    Label2: TLabel;
    Label3: TLabel;
    procedure
BubbleSortBoxPaint(Sender:
TObject);
    procedure
SelectionSortBoxPaint(Sender:
TObject);
    procedure
QuickSortBoxPaint(Sender:
TObject);
    procedure FormCreate(Sender:
TObject);
    procedure StartBtnClick(Sender:
TObject);
  private
    ThreadsRunning: Integer;
    procedure RandomizeArrays;
    procedure ThreadDone(Sender:
TObject);
  public
    procedure PaintArray(Box:
TPaintBox; const A: array of
Integer);
  end;

var
  ThreadSortForm:
TThreadSortForm;

implementation

uses SortThds;

{$R *.dfm}

type
  PSortArray = ^TSortArray;
  TSortArray =  array[0..114] of
Integer;

var
  ArraysRandom: Boolean;
  BubbleSortArray,
SelectionSortArray,
QuickSortArray: TSortArray;

{ TThreadSortForm }

procedure
TThreadSortForm.PaintArray(Box:
TPaintBox; const A: array of
Integer);
var
  I: Integer;
begin
  with Box do
  begin
    Canvas.Pen.Color := clRed;
    for I := Low(A) to High(A) do
PaintLine(Canvas, I, A[I]);
  end;
end;
```

```
    OnTerminate := ThreadDone;
  StartBtn.Enabled := False;
end;

procedure TThreadSortForm.RandomizeArrays;
var
  I: Integer;
begin
  if not ArraysRandom then
  begin
    Randomize;
    for I := Low(BubbleSortArray) to High(BubbleSortArray) do
      BubbleSortArray[I] := Random(170);
    SelectionSortArray := BubbleSortArray;
    QuickSortArray := BubbleSortArray;
    ArraysRandom := True;
    Repaint;
  end;
end;

procedure TThreadSortForm.ThreadDone(Sender: TObject);
begin
  Dec(ThreadsRunning);
  if ThreadsRunning = 0 then
  begin
    StartBtn.Enabled := True;
    ArraysRandom := False;
  end;
end;

end.

\\\\\\\\\\\\\\\\\\


procedure TThreadSortForm.BubbleSortBoxPaint(Sender: TObject);
begin
  PaintArray(BubbleSortBox, BubbleSortArray);
end;

procedure TThreadSortForm.SelectionSortBoxPaint(Sender: TObject);
begin
  PaintArray(SelectionSortBox, SelectionSortArray);
end;

procedure TThreadSortForm.QuickSortBoxPaint(Sender: TObject);
begin
  PaintArray(QuickSortBox, QuickSortArray);
end;

procedure TThreadSortForm.FormCreate(Sender: TObject);
begin
  RandomizeArrays;
end;

procedure TThreadSortForm.StartBtnClick(Sender: TObject);
begin
  RandomizeArrays;
  ThreadsRunning := 3;
  with TBubbleSort.Create(BubbleSortBox, BubbleSortArray) do
    OnTerminate := ThreadDone;
  with TSelectionSort.Create(SelectionSortBox, SelectionSortArray) do
    OnTerminate := ThreadDone;
  with TQuickSort.Create(QuickSortBox, QuickSortArray) do
```

```
  Left = 192
  Top = 24
  Width = 177
  Height = 233
 end
 object BubbleSortBox: TPaintBox
  Left = 8
  Top = 24
  Width = 177
  Height = 233
  OnPaint = BubbleSortBoxPaint
 end
 object SelectionSortBox:
TPaintBox
  Left = 192
  Top = 24
  Width = 177
  Height = 233
  OnPaint = SelectionSortBoxPaint
 end
 object QuickSortBox: TPaintBox
  Left = 376
  Top = 24
  Width = 177
  Height = 233
  OnPaint = QuickSortBoxPaint
 end
 object Label1: TLabel
  Left = 8
  Top = 8
  Width = 55
  Height = 13
  Caption = 'Bubble Sort'
 end
 object Label2: TLabel
  Left = 192
  Top = 8
  Width = 66
  Height = 13
  Caption = 'Selection Sort'
 end
 object Label3: TLabel
  Left = 376
  Top = 8
  Width = 50
  Height = 13
  Caption = 'Quick Sort'
 end
```

ObjectS
الملف لبرنامج
(Threads)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
object ThreadSortForm:
TThreadSortForm
 Left = 182
 Top = 115
 BorderStyle = bsDialog
 Caption = 'Thread Sorting Demo'
 ClientHeight = 295
 ClientWidth = 562
 Color = clBtnFace
 Font.Charset =
DEFAULT_CHARSET
 Font.Color = clWindowText
 Font.Height = -11
 Font.Name = 'MS Sans Serif'
 Font.Style = []
 OldCreateOrder = True
 Position = poScreenCenter
 OnCreate = FormCreate
 PixelsPerInch = 96
 TextHeight = 13
 object Bevel1: TBevel
  Left = 8
  Top = 24
  Width = 177
  Height = 233
 end
 object Bevel3: TBevel
  Left = 376
  Top = 24
  Width = 177
  Height = 233
 end
 object Bevel2: TBevel
```

```
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

<-------> procedure
TForm1.FormCreate(Sender:
TObject);
begin
  Listbox1.Items := Screen.Fonts;
end;

<-------> procedure
TForm1.ListBox1Click(Sender:
TObject);
begin
  FontLabel.Caption :=
ListBox1.Items[ListBox1.ItemIndex]
;
end;

<-------> procedure
TForm1.DrawItem(Control:
TWinControl; Index: Integer;
  Rect: TRect; State:
TOwnerDrawState);
begin
  with ListBox1.Canvas do
  begin
    FillRect(Rect);
    Font.Name :=
ListBox1.Items[Index];
    Font.Size := 0;    // use font's
preferred size
    TextOut(Rect.Left+1, Rect.Top+1,
ListBox1.Items[Index]);
  end;
end;

<-------> procedure
TForm1.ListBox1MeasureItem(Cont
rol: TWinControl; Index: Integer;
  var Height: Integer);
```

```
  object StartBtn: TButton
    Left = 480
    Top = 264
    Width = 75
    Height = 25
    Caption = 'Start Sorting'
    TabOrder = 0
    OnClick = StartBtnClick
  end
end
```

الملف التنفيذي لبرنامج

8 procedures

Ownerlst)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit Fontlist;

interface

uses Windows, Classes, Graphics,
Forms, Controls, StdCtrls;

type
  TForm1 = class(TForm)
    ListBox1: TListBox;
    Label1: TLabel;
    FontLabel: TLabel;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
ListBox1Click(Sender: TObject);
    <-------> procedure
DrawItem(Control: TWinControl;
Index: Integer; Rect: TRect;
      State: TOwnerDrawState);
    <-------> procedure
ListBox1MeasureItem(Control:
TWinControl; Index: Integer;
      var Height: Integer);
  private
    { Private declarations }
```

```
    Height = 16
    Caption = 'System Fonts'
  end
  <-------> object FontLabel: TLabel
    Left = 20
    Top = 305
    Width = 241
    Height = 20
    AutoSize = False
  end
  <-------> object ListBox1: TListBox
    Left = 20
    Top = 45
    Width = 241
    Height = 251
    Style = lbOwnerDrawVariable
    ItemHeight = 16
    TabOrder = 0
    OnClick = ListBox1Click
    OnDrawItem = DrawItem
    OnMeasureItem =
ListBox1MeasureItem
  end
end
```

```
begin
  with ListBox1.Canvas do
  begin
    Font.Name :=
Listbox1.Items[Index];
    Font.Size := 0;              // use
font's preferred size
    Height := TextHeight('Wg') + 2; //
measure ascenders and descenders
  end;
end;

end.
```

<-------> objectS4

الملف النصي لبرنامج

(---Ownerlst)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<-------> object Form1: TForm1
  Left = 174
  Top = 108
  ActiveControl = ListBox1
  BorderStyle = bsDialog
  Caption = 'OwnerDraw'
  ClientHeight = 359
  ClientWidth = 276
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Scaled = False
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 16
  <-------> object Label1: TLabel
    Left = 20
    Top = 20
    Width = 81
```

الملف التنفيذي لبرنامج
procedure50
(---CoolStuf)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١
unit Main;
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, ExtCtrls, Menus, ComCtrls, OleCtrls, Buttons, ToolWin, Isp3,
ActnList, ImgList, SHDocVw;
const
CM_HOMEPAGEREQUEST = WM_USER + $1000;
type
TMainForm = class(TForm)
StatusBar1: TStatusBar;
MainMenu1: TMainMenu;
File1: TMenuItem;
Exit1: TMenuItem;
View1: TMenuItem;
NavigatorImages: TImageList;
NavigatorHotImages: TImageList;
LinksImages: TImageList;
LinksHotImages: TImageList;
CoolBar1: TCoolBar;
ToolBar1: TToolBar;
BackBtn: TToolButton;
ForwardBtn: TToolButton;

StopBtn: TToolButton;
RefreshBtn: TToolButton;
ToolBar2: TToolBar;
ToolButton6: TToolButton;
ToolButton5: TToolButton;
ToolButton7: TToolButton;
ToolButton8: TToolButton;
Animate1: TAnimate;
URLs: TComboBox;
Help1: TMenuItem;
About1: TMenuItem;
Toolbar3: TMenuItem;
Statusbar2: TMenuItem;
Go1: TMenuItem;
Back1: TMenuItem;
Forward1: TMenuItem;
Stop1: TMenuItem;
Refresh1: TMenuItem;
N2: TMenuItem;
ActionList1: TActionList;
BackAction: TAction;
ForwardAction: TAction;
StopAction: TAction;
RefreshAction: TAction;
WebBrowser1: TWebBrowser;
<-------> procedure Exit1Click(Sender: TObject);
<-------> procedure About1Click(Sender: TObject);
<-------> procedure StopClick(Sender: TObject);
<-------> procedure URLsKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
<-------> procedure FormCreate(Sender: TObject);
<-------> procedure LinksClick(Sender: TObject);
<-------> procedure RefreshClick(Sender: TObject);
<-------> procedure BackClick(Sender: TObject);
<-------> procedure ForwardClick(Sender: TObject);
<-------> procedure FormDestroy(Sender: TObject);
<-------> procedure URLsClick(Sender: TObject);

```
TMainForm.Exit1Click(Sender: TObject);
begin
  Close;
end;

<-------> procedure TMainForm.FindAddress;
var
  Flags: OLEVariant;

begin
  Flags := 0;
  UpdateCombo := True;
  WebBrowser1.Navigate(WideString(Urls.Text), Flags, Flags, Flags, Flags);
end;

<-------> procedure TMainForm.About1Click(Sender: TObject);
begin
  ShowAboutBox;
end;

<-------> procedure TMainForm.StopClick(Sender: TObject);
begin
  WebBrowser1.Stop;
end;

<-------> procedure TMainForm.URLsKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key = VK_Return then
  begin
    FindAddress;
  end;
end;

<-------> procedure TMainForm.URLsClick(Sender: TObject);
begin
```

```
    <-------> procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    <-------> procedure Toolbar3Click(Sender: TObject);
    <-------> procedure Statusbar2Click(Sender: TObject);
    <-------> procedure BackActionUpdate(Sender: TObject);
    <-------> procedure ForwardActionUpdate(Sender: TObject);
    <-------> procedure WebBrowser1BeforeNavigate2(Sender: TObject;
      const pDisp: IDispatch; var URL, Flags, TargetFrameName, PostData,
      Headers: OleVariant; var Cancel: WordBool);
    <-------> procedure WebBrowser1DownloadBegin(Sender: TObject);
    <-------> procedure WebBrowser1DownloadComplete(Sender: TObject);
  private
    HistoryIndex: Integer;
    HistoryList: TStringList;
    UpdateCombo: Boolean;
    <-------> procedure FindAddress;
    <-------> procedure HomePageRequest(var message: tmessage); message CM_HOMEPAGEREQUEST;
  end;

var
  MainForm: TMainForm;

implementation

uses About;

{$R *.dfm}

<-------> procedure
```

```
BackBtn.Enabled then
    BackBtn.Click;
end;

<-------> procedure
TMainForm.Toolbar3Click(Sender:
TObject);
begin
  with Sender as TMenuItem do
  begin
    Checked := not Checked;
    Coolbar1.Visible := Checked;
  end;
end;

<-------> procedure
TMainForm.Statusbar2Click(Sende
r: TObject);
begin
  with Sender as TMenuItem do
  begin
    Checked := not Checked;
    StatusBar1.Visible := Checked;
  end;
end;

<-------> procedure
TMainForm.HomePageRequest(var
Message: TMessage);
begin
  URLs.Text :=
'http://www.borland.com';
  FindAddress;
end;

<-------> procedure
TMainForm.FormCreate(Sender:
TObject);
begin
  HistoryIndex := -1;
  HistoryList := TStringList.Create;
  { Load the animation from the AVI
file in the startup directory.  An
    alternative to this would be to
create a .RES file including the
cool.avi
    as an AVI resource and use the
ResName or ResId properties of
Animate1 to
```

```
  FindAddress;
end;

<-------> procedure
TMainForm.LinksClick(Sender:
TObject);
begin
  if (Sender as TToolButton).Hint = ''
then Exit;
  URLs.Text := (Sender as
TToolButton).Hint;
  FindAddress;
end;

<-------> procedure
TMainForm.RefreshClick(Sender:
TObject);
begin
  FindAddress;
end;

<-------> procedure
TMainForm.BackClick(Sender:
TObject);
begin
  URLs.Text :=
HistoryList[HistoryIndex - 1];
  FindAddress;
end;

<-------> procedure
TMainForm.ForwardClick(Sender:
TObject);
begin
  URLs.Text :=
HistoryList[HistoryIndex + 1];
  FindAddress;
end;

<-------> procedure
TMainForm.FormKeyDown(Sender
: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Shift = [ssAlt] then
    if (Key = VK_RIGHT) and
ForwardBtn.Enabled then
      ForwardBtn.Click
    else if (Key = VK_LEFT) and
```

```
var
  NewIndex: Integer;
begin
  NewIndex :=
HistoryList.IndexOf(URL);
  if NewIndex = -1 then
  begin
    { Remove entries in HistoryList
between last address and current
address }
    if (HistoryIndex >= 0) and
(HistoryIndex < HistoryList.Count -
1) then
      while HistoryList.Count >
HistoryIndex do

HistoryList.Delete(HistoryIndex);
    HistoryIndex :=
HistoryList.Add(URL);
  end
  else
    HistoryIndex := NewIndex;
  if UpdateCombo then
  begin
    UpdateCombo := False;
    NewIndex :=
URLs.Items.IndexOf(URL);
    if NewIndex = -1 then
      URLs.Items.Insert(0, URL)
    else
      URLs.Items.Move(NewIndex, 0);
  end;
  URLs.Text := URL;
  Statusbar1.Panels[0].Text := URL;
end;

<-------> procedure
TMainForm.WebBrowser1Downloa
dBegin(Sender: TObject);
begin
  { Turn the stop button dark red }
  StopBtn.ImageIndex := 4;
  { Play the avi from the first frame
indefinitely }
  Animate1.Active := True;
end;

<-------> procedure
TMainForm.WebBrowser1Downloa
```

```
    point to it. }
  Animate1.FileName :=
ExtractFilePath(Application.ExeNa
me) + 'cool.avi';
  { Find the home page - needs to be
posted because HTML control
hasn't been
    registered yet. }
  PostMessage(Handle,
CM_HOMEPAGEREQUEST, 0, 0);
end;

<-------> procedure
TMainForm.FormDestroy(Sender:
TObject);
begin
  HistoryList.Free;
end;

<-------> procedure
TMainForm.BackActionUpdate(Sen
der: TObject);
begin
  if HistoryList.Count > 0 then
    BackAction.Enabled :=
HistoryIndex > 0
  else
    BackAction.Enabled := False;
end;

<-------> procedure
TMainForm.ForwardActionUpdate(
Sender: TObject);
begin
  if HistoryList.Count > 0 then
    ForwardAction.Enabled :=
HistoryIndex < HistoryList.Count -
1
  else
    ForwardAction.Enabled := False;
end;

<-------> procedure
TMainForm.WebBrowser1BeforeNa
vigate2(Sender: TObject;
  const pDisp: IDispatch; var URL,
Flags, TargetFrameName, PostData,
  Headers: OleVariant; var Cancel:
WordBool);
```

```
<-------> procedure ShowAboutBox;
begin
  with TAboutBox.Create(Application) do
    try
      ShowModal;
    finally
      Free;
    end;
end;

<-------> procedure TAboutBox.GetOSInfo;
var
  Platform: string;
  BuildNumber: Integer;
begin
  case Win32Platform of
    VER_PLATFORM_WIN32_WINDOWS:
      begin
        Platform := 'Windows 95';
        BuildNumber := Win32BuildNumber and $0000FFFF;
      end;
    VER_PLATFORM_WIN32_NT:
      begin
        Platform := 'Windows NT';
        BuildNumber := Win32BuildNumber;
      end;
      else
      begin
        Platform := 'Windows';
        BuildNumber := 0;
      end;
  end;
  if (Win32Platform = VER_PLATFORM_WIN32_WINDOWS) or
    (Win32Platform = VER_PLATFORM_WIN32_NT) then
  begin
    if Win32CSDVersion = '' then
      OS.Caption := Format('%s %d.%d (Build %d)', [Platform,
```

```
dComplete(Sender: TObject);
begin
  { Turn the stop button grey }
  StopBtn.ImageIndex := 2;
  { Stop the avi and show the first frame }
  Animate1.Active := False;
end;

end.
\\\\\\\\\\\
unit About;

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, Buttons, ExtCtrls;

type
  TAboutBox = class(TForm)
    OKButton: TButton;
    Copyright: TLabel;
    Bevel1: TBevel;
    SKUName: TLabel;
    Image2: TImage;
    PhysMem: TLabel;
    OS: TLabel;
    Label3: TLabel;
    Image1: TImage;
    <-------> procedure FormCreate(Sender: TObject);
  private
    <-------> procedure GetOSInfo;
    <-------> procedure InitializeCaptions;
  end;

<-------> procedure ShowAboutBox;

implementation

uses ShellAPI;

{$R *.dfm}

```

علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
<-------> object MainForm: TMainForm
  Left = 173
  Top = 148
  Width = 503
  Height = 370
  Caption = 'Web Browser'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  Menu = MainMenu1
  OldCreateOrder = True
  Position = poScreenCenter
  Scaled = False
  ShowHint = True
  Visible = True
  OnCreate = FormCreate
  OnDestroy = FormDestroy
  OnKeyDown = FormKeyDown
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object StatusBar1: TStatusBar
    Left = 0
    Top = 297
    Width = 495
    Height = 19
    Panels = <
      item
        Width = 50
      end>
  end
  <-------> object CoolBar1: TCoolBar
    Left = 0
    Top = 0
    Width = 495
    Height = 91
    AutoSize = True
    Bands = <
```

```
Win32MajorVersion,
      Win32MinorVersion, BuildNumber])
    else
     OS.Caption := Format('%s %d.%d (Build %d: %s)', [Platform, Win32MajorVersion,
      Win32MinorVersion, BuildNumber, Win32CSDVersion]);
  end
  else
    OS.Caption := Format('%s %d.%d', [Platform, Win32MajorVersion,
     Win32MinorVersion])
end;

<-------> procedure TAboutBox.InitializeCaptions;
var
  MS: TMemoryStatus;
begin
  GetOSInfo;
  MS.dwLength := SizeOf(TMemoryStatus);
  GlobalMemoryStatus(MS);
  PhysMem.Caption := FormatFloat('#,###" KB"', MS.dwTotalPhys div 1024);
end;

<-------> procedure TAboutBox.FormCreate(Sender: TObject);
begin
  InitializeCaptions;
  with Image2.Picture.Bitmap do
    Handle := CreateGrayMappedBmp(Handle);
end;
end.
```

Objects 46

الملف النصي لبرنامج

(--CoolStuf)

إعداد

```
    ShowCaptions = True
    TabOrder = 0
    Transparent = True
    Wrapable = False
    <-------> object BackBtn: TToolButton
      Left = 0
      Top = 0
      Action = BackAction
    end
    <-------> object ForwardBtn: TToolButton
      Left = 45
      Top = 0
      Action = ForwardAction
    end
    <-------> object StopBtn: TToolButton
      Left = 90
      Top = 0
      Action = StopAction
    end
    <-------> object RefreshBtn: TToolButton
      Left = 135
      Top = 0
      Action = RefreshAction
    end
  end
  <-------> object ToolBar2: TToolBar
    Left = 38
    Top = 65
    Width = 449
    Height = 22
    AutoSize = True
    ButtonWidth = 94
    EdgeBorders = []
    Flat = True
    HotImages = LinksHotImages
    Images = LinksImages
    List = True
    ShowCaptions = True
    TabOrder = 1
    Transparent = True
    Wrapable = False
    <-------> object ToolButton6: TToolButton
```

```
    item
      Break = False
      Control = ToolBar1
      ImageIndex = -1
      MinHeight = 40
      Width = 429
    end
    item
      Break = False
      Control = Animate1
      FixedSize = True
      ImageIndex = -1
      MinHeight = 39
      MinWidth = 60
      Width = 60
    end
    item
      Control = URLs
      ImageIndex = -1
      MinHeight = 21
      Text = 'Address'
      Width = 491
    end
    item
      Control = ToolBar2
      ImageIndex = -1
      MinHeight = 22
      Text = 'Links'
      Width = 491
    end>
  ParentShowHint = False
  Bitmap.Data = {
    }
  ShowHint = False
  <-------> object ToolBar1: TToolBar
    Left = 9
    Top = 0
    Width = 416
    Height = 40
    AutoSize = True
    ButtonHeight = 40
    ButtonWidth = 45
    EdgeBorders = []
    Flat = True
    HotImages = NavigatorHotImages
    Images = NavigatorImages
```

```
  end
  <-------> object URLs: TComboBox
    Left = 51
    Top = 42
    Width = 436
    Height = 21
    Hint = 'URL'
    ItemHeight = 13
    TabOrder = 3
    OnClick = URLsClick
    OnKeyDown = URLsKeyDown
  end
 end
 <-------> object WebBrowser1: TWebBrowser
  Left = 0
  Top = 91
  Width = 495
  Height = 206
  Align = alClient
  TabOrder = 2
  OnDownloadBegin = WebBrowser1DownloadBegin
  OnDownloadComplete = WebBrowser1DownloadComplete
  OnBeforeNavigate2 = WebBrowser1BeforeNavigate2
  ControlData = {
    }
 end
 <-------> object MainMenu1: TMainMenu
  Left = 128
  Top = 200
  <-------> object File1: TMenuItem
    Caption = '&File'
    <-------> object Exit1: TMenuItem
      Caption = 'E&xit'
      OnClick = Exit1Click
    end
  end
  <-------> object View1: TMenuItem
    Caption = '&View'
    <-------> object Toolbar3: TMenuItem
```

```
        Left = 0
        Top = 0
        Hint = 'http://www.informant.com:80/delphi/'
        Caption = 'Informant'
        ImageIndex = 0
        OnClick = LinksClick
      end
      <-------> object ToolButton5: TToolButton
        Left = 94
        Top = 0
        Hint = 'http://www.delphi-jedi.org/'
        Caption = 'Delphi Jedi'
        ImageIndex = 1
        OnClick = LinksClick
      end
      <-------> object ToolButton7: TToolButton
        Left = 188
        Top = 0
        Hint = 'http://www.borland.com:80/delphi'
        Caption = 'Product News'
        ImageIndex = 2
        OnClick = LinksClick
      end
      <-------> object ToolButton8: TToolButton
        Left = 282
        Top = 0
        Hint = 'http://www.borland.com:80'
        Caption = 'Borland'
        ImageIndex = 3
        OnClick = LinksClick
      end
    end
    <-------> object Animate1: TAnimate
      Left = 431
      Top = 0
      Width = 60
      Height = 39
      FileName = 'Cool.avi'
      StopFrame = 13
```

```
    Top = 104
    Bitmap = {

  end
  <-------> object ActionList1:
TActionList
    Images = NavigatorImages
    Left = 80
    Top = 160
    <-------> object BackAction:
TAction
      Caption = 'Back'
      ImageIndex = 0
      OnExecute = BackClick
      OnUpdate = BackActionUpdate
    end
    <-------> object ForwardAction:
TAction
      Caption = 'Forward'
      Enabled = False
      ImageIndex = 1
      OnExecute = ForwardClick
      OnUpdate =
ForwardActionUpdate
    end
    <-------> object StopAction:
TAction
      Caption = 'Stop'
      ImageIndex = 2
      OnExecute = StopClick
    end
    <-------> object RefreshAction:
TAction
      Caption = 'Refresh'
      ImageIndex = 3
      OnExecute = RefreshClick
    end
  end
End
\\\\\\\\\\\\\\\\
<-------> object AboutBox:
TAboutBox
  Left = 142
  Top = 116
  BorderStyle = bsDialog
  Caption = 'About Web Browser'
  ClientHeight = 258
  ClientWidth = 342
```

```
        Caption = '&Toolbar'
        Checked = True
        OnClick = Toolbar3Click
      end
      <-------> object Statusbar2:
TMenuItem
        Caption = 'Status &Bar'
        Checked = True
        OnClick = Statusbar2Click
      end
      <-------> object N2: TMenuItem
        Caption = '-'
      end
      <-------> object Stop1:
TMenuItem
        Action = StopAction
      end
      <-------> object Refresh1:
TMenuItem
        Action = RefreshAction
      end
    end
    <-------> object Go1: TMenuItem
      Caption = '&Go'
      <-------> object Back1:
TMenuItem
        Action = BackAction
      end
      <-------> object Forward1:
TMenuItem
        Action = ForwardAction
      end
    end
    <-------> object Help1:
TMenuItem
      Caption = '&Help'
      <-------> object About1:
TMenuItem
        Caption = '&About...'
        OnClick = About1Click
      end
    end
  end
  <-------> object NavigatorImages:
TImageList
    Height = 20
    Width = 30
    Left = 32
```

```
Color = clBtnFace
ParentFont = True
OldCreateOrder = True
Position = poScreenCenter
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> object Copyright: TLabel
  Left = 82
  Top = 81
  Width = 207
  Height = 13
  Caption = 'Copyright 1983 - 2002 Borland International'
  Font.Charset = ANSI_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
end
<-------> object Bevel1: TBevel
  Left = 86
  Top = 158
  Width = 251
  Height = 2
end
<-------> object SKUName: TLabel
  Left = 82
  Top = 57
  Width = 97
  Height = 16
  Caption = 'Using Delphi 7.0'
  Font.Charset = ANSI_CHARSET
  Font.Color = clBlack
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
end
<-------> object Image2: TImage
  Left = 76
  Top = 13
  Width = 170
  Height = 35
  AutoSize = True
  Picture.Data = {
    000}
  Transparent = True
end
<-------> object Label3: TLabel
  Left = 86
  Top = 184
  Width = 145
  Height = 13
  Caption = 'Memory Available to Windows:'
end
<-------> object PhysMem: TLabel
  Left = 323
  Top = 184
  Width = 6
  Height = 13
  Alignment = taRightJustify
  Caption = '0'
end
<-------> object OS: TLabel
  Left = 86
  Top = 167
  Width = 15
  Height = 13
  Caption = 'OS'
end
<-------> object Image1: TImage
  Left = 9
  Top = 11
  Width = 60
  Height = 39
  AutoSize = True
  Picture.Data = {

    0000}
  Transparent = True
end
<-------> object OKButton: TButton
  Left = 256
  Top = 223
  Width = 75
  Height = 25
  Cancel = True
  Caption = 'OK'
  Default = True
  ModalResult = 1
  TabOrder = 0
end
end
```

```
Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, jpeg, ExtCtrls, FileCtrl, ComCtrls, Menus, printers;

type
  TForm1 = class(TForm)
    Panel1: TPanel;
    DirectoryListBox1: TDirectoryListBox;
    FileListBox1: TFileListBox;
    Panel3: TPanel;
    DriveComboBox1: TDriveComboBox;
    Scale: TComboBox;
    PixelFormat: TComboBox;
    ColorSpace: TComboBox;
    Performance: TComboBox;
    ProgressiveDisplay: TCheckBox;
    IncrementalDisplay: TCheckBox;
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    Open1: TMenuItem;
    N1: TMenuItem;
    Print1: TMenuItem;
    PrinterSetup1: TMenuItem;
    N2: TMenuItem;
    Exit1: TMenuItem;
    OpenDialog1: TOpenDialog;
    PrinterSetupDialog1: TPrinterSetupDialog;
    PrintDialog1: TPrintDialog;
    ScrollBox1: TScrollBox;
    Image1: TImage;
    <------>procedure FileListBox1DblClick(Sender: TObject);
    <------>procedure SetJPEGOptions(Sender: TObject);
    <------>procedure FormCreate(Sender: TObject);
    <------>procedure ProgressUpdate(Sender: TObject; Stage: TProgressStage;
      PercentDone: Byte; RedrawNow: Boolean; const R: TRect; const Msg: string);
    <------>procedure Open1Click(Sender: TObject);
```

القسم الرابع
المستكشف في دلفي

Procedures20
الملف التنفيذي لبرنامج
(Jpeg)
إعداد **علاء الدين اللباد**
هاتف ٠٩٤٤٥٧٥٣٧١

```
program jpegproj;

uses
  Forms,
  jpegdemo in 'jpegdemo.pas' {Form1};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.Run;
end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

unit jpegdemo;

interface

uses
  Windows, Messages, SysUtils,
```

```
var
  Temp: Boolean;
begin
  Temp := Image1.Picture.Graphic is
TJPEGImage;
  if Temp then
    with
TJPEGImage(Image1.Picture.Grap
hic) do
    begin
      PixelFormat :=
TJPEGPixelFormat(Self.PixelForma
t.ItemIndex);
      Scale :=
TJPEGScale(Self.Scale.ItemIndex);
      Grayscale :=
Boolean(Colorspace.ItemIndex);
      Performance :=
TJPEGPerformance(Self.Performan
ce.ItemIndex);
      ProgressiveDisplay :=
Self.ProgressiveDisplay.Checked;
    end;
  Scale.Enabled := Temp;
  PixelFormat.Enabled := Temp;
  Colorspace.Enabled := Temp;
  Performance.Enabled := Temp;
  ProgressiveDisplay.Enabled :=
Temp
    and
TJPEGImage(Image1.Picture.Grap
hic).ProgressiveEncoding;
  Image1.IncrementalDisplay :=
IncrementalDisplay.Checked;
end;

<------>procedure
TForm1.FormCreate(Sender:
TObject);
begin
  Scale.ItemIndex := 0;
  PixelFormat.ItemIndex := 0;
  Colorspace.ItemIndex := 0;
  Performance.ItemIndex := 0;
  OpenDialog1.Filter :=
GraphicFilter(TGraphic);
  FileListbox1.Mask :=
GraphicFileMask(TGraphic);
  Image1.OnProgress :=
```

```
    <------>procedure
Print1Click(Sender: TObject);
    <------>procedure
PrinterSetup1Click(Sender:
TObject);
    <------>procedure
Exit1Click(Sender: TObject);
    <------>procedure
ScrollBox1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    <------>procedure  OpenFile(const
Filename: string);
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

<------>procedure
TForm1.OpenFile(const Filename:
string);
begin
  try

Image1.Picture.LoadFromFile(Filen
ame);
  except
    on EInvalidGraphic do
      Image1.Picture.Graphic := nil;
  end;
  SetJPEGOptions(self);
end;

<------>procedure
TForm1.FileListBox1DblClick(Send
er: TObject);
begin
  OpenFile(FileListbox1.Filename);
end;

<------>procedure
TForm1.SetJPEGOptions(Sender:
TObject);
```

```
  begin
    if OutputWidth < OutputHeight
then
    begin
      OutputHeight :=
Printer.PageHeight;
      OutputWidth := OutputHeight
* AspectRatio;
    end
    else
    begin
      OutputWidth :=
Printer.PageWidth;
      OutputHeight := OutputWidth /
AspectRatio;
    end
  end;
  if OutputWidth >
Printer.PageWidth then
  begin
    OutputWidth :=
Printer.PageWidth;
    OutputHeight := OutputWidth /
AspectRatio;
  end;
  if OutputHeight >
Printer.PageHeight then
  begin
    OutputHeight :=
Printer.PageHeight;
    OutputWidth := OutputHeight *
AspectRatio;
  end;

Printer.Canvas.StretchDraw(Rect(0,
0,
    Trunc(OutputWidth),
Trunc(OutputHeight)),
    Image1.Picture.Graphic);
 finally
  Printer.EndDoc;
 end;
end;

<------>procedure
TForm1.PrinterSetup1Click(Sender:
TObject);
begin
 PrinterSetupDialog1.Execute;
```

```
ProgressUpdate;
end;

<------>procedure
TForm1.ProgressUpdate(Sender:
TObject; Stage: TProgressStage;
 PercentDone: Byte; RedrawNow:
Boolean; const R: TRect; const Msg:
string);
begin
 if Stage = psRunning then
  Caption :=
Format('%d%%',[PercentDone])
 else
  Caption := 'Form1';
end;


<------>procedure
TForm1.Open1Click(Sender:
TObject);
begin
 if OpenDialog1.Execute then

OpenFile(OpenDialog1.FileName);
end;

<------>procedure
TForm1.Print1Click(Sender:
TObject);
var
 AspectRatio: Single;
 OutputWidth, OutputHeight:
Single;
begin
 if not PrintDialog1.Execute then
Exit;
 Printer.BeginDoc;
 try
  OutputWidth :=
Image1.Picture.Width;
  OutputHeight :=
Image1.Picture.Height;
  AspectRatio := OutputWidth /
OutputHeight;
  if (OutputWidth <
Printer.PageWidth) and
    (OutputHeight <
Printer.PageHeight) then
```

```
    Top = 27
    Width = 124
    Height = 349
    Align = alLeft
    TabOrder = 0
    <------>Object DirectoryListBox1:
TDirectoryListBox
      Left = 1
      Top = 1
      Width = 122
      Height = 79
      Align = alTop
      Ctl3D = True
      FileList = FileListBox1
      ItemHeight = 16
      ParentCtl3D = False
      TabOrder = 0
    end
    <------>Object FileListBox1:
TFileListBox
      Left = 1
      Top = 80
      Width = 122
      Height = 268
      Align = alClient
      ItemHeight = 13
      TabOrder = 1
      OnClick = FileListBox1DblClick
    end
  end
  <------>Object Panel3: TPanel
    Left = 0
    Top = 0
    Width = 488
    Height = 27
    Align = alTop
    TabOrder = 1
    <------>Object DriveComboBox1:
TDriveComboBox
      Left = 7
      Top = 3
      Width = 117
      Height = 19
      DirList = DirectoryListBox1
      TabOrder = 0
    end
    <------>Object Scale: TComboBox
      Left = 127
```

```
end;

<------>procedure
TForm1.Exit1Click(Sender:
TObject);
begin
  Close;
end;

<------>procedure
TForm1.ScrollBox1Click(Sender:
TObject);
begin

end;
```

<------>ObjectS26

الملف النصي لبرنامج

(Jpeg)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<------>Object Form1: TForm1
  Left = 225
  Top = 115
  AutoScroll = False
  Caption = 'Form1'
  ClientHeight = 376
  ClientWidth = 488
  Color = clBtnFace
  Font.Charset = ANSI_CHARSET
  Font.Color = clWindowText
  Font.Height = -10
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  Menu = MainMenu1
  OldCreateOrder = True
  Position = poDefaultPosOnly
  ShowHint = True
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <------>Object Panel1: TPanel
    Left = 0
```

```
TComboBox
    Left = 283
    Top = 3
    Width = 75
    Height = 21
    Hint = 'Performance'
    Style = csDropDownList
    Enabled = False
    ItemHeight = 13
    TabOrder = 4
    OnClick = SetJPEGOptions
    Items.Strings = (
      'Quality'
      'Speed')
  end
  <------>Object ProgressiveDisplay:
TCheckBox
    Left = 364
    Top = 1
    Width = 124
    Height = 14
    Caption = 'Progressive JPEG'
    Enabled = False
    Font.Charset =
ANSI_CHARSET
    Font.Color = clWindowText
    Font.Height = -9
    Font.Name = 'arial'
    Font.Style = []
    ParentFont = False
    TabOrder = 5
    OnClick = SetJPEGOptions
  end
  <------>Object
IncrementalDisplay: TCheckBox
    Left = 364
    Top = 12
    Width = 137
    Height = 14
    Caption = 'Incremental Display'
    Font.Charset =
ANSI_CHARSET
    Font.Color = clWindowText
    Font.Height = -9
    Font.Name = 'Arial'
    Font.Style = []
    ParentFont = False
    TabOrder = 6
    Top = 3
    Width = 36
    Height = 21
    Hint = 'Scale'
    Style = csDropDownList
    Enabled = False
    ItemHeight = 13
    TabOrder = 1
    OnClick = SetJPEGOptions
    Items.Strings = (
      '1:1'
      '1:2'
      '1:4'
      '1:8')
  end
  <------>Object PixelFormat:
TComboBox
    Left = 166
    Top = 3
    Width = 49
    Height = 21
    Hint = 'Pixel Format'
    Style = csDropDownList
    Enabled = False
    ItemHeight = 13
    TabOrder = 2
    OnClick = SetJPEGOptions
    Items.Strings = (
      '24 bit'
      '8 bit')
  end
  <------>Object ColorSpace:
TComboBox
    Left = 218
    Top = 3
    Width = 62
    Height = 21
    Hint = 'Color space'
    Style = csDropDownList
    Enabled = False
    ItemHeight = 13
    TabOrder = 3
    OnClick = SetJPEGOptions
    Items.Strings = (
      'RGB'
      'Grayscale')
  end
  <------>Object Performance:
```

ShortCut = 32856
OnClick = Exit1Click
end
end
end
<------>Object OpenDialog1: TOpenDialog
Options = [ofPathMustExist, ofFileMustExist]
Left = 160
Top = 225
end
<------>Object PrinterSetupDialog1: TPrinterSetupDialog
Left = 168
Top = 305
end
<------>Object PrintDialog1: TPrintDialog
Left = 96
Top = 377
end
end

end
end
<------>Object ScrollBox1: TScrollBox
Left = 124
Top = 27
Width = 364
Height = 349
Align = alClient
TabOrder = 2
<------>Object Image1: TImage
Left = 0
Top = 0
Width = 85
Height = 85
AutoSize = True
end
end
<------>Object MainMenu1: TMainMenu
Left = 152
Top = 145
<------>Object File1: TMenuItem
Caption = '&File'
<------>Object Open1: TMenuItem
Caption = '&Open...'
ShortCut = 114
OnClick = Open1Click
end
<------>Object N1: TMenuItem
Caption = '-'
end
<------>Object Print1: TMenuItem
Caption = '&Print'
OnClick = Print1Click
end
<------>Object PrinterSetup1: TMenuItem
Caption = 'Printer &Setup...'
OnClick = PrinterSetup1Click
end
<------>Object N2: TMenuItem
Caption = '-'
end
<------>Object Exit1: TMenuItem
Caption = 'E&xit'

الملف التنفيذي لبرنامج

Virtual Listview

Procedure40

)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit VListView;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  ComCtrls, StdCtrls, ToolWin,
ShlObj, ImgList, Menus;

type
  PShellItem = ^TShellItem;
  TShellItem = record
    FullID,
    ID: PItemIDList;
    Empty: Boolean;
    DisplayName,
    TypeName: string;
    ImageIndex,
    Size,
    Attributes: Integer;
    ModDate: string;
  end;

  TForm1 = class(TForm)
    ListView: TListView;
    CoolBar1: TCoolBar;
    ToolBar2: TToolBar;
    ToolbarImages: TImageList;
    btnBrowse: TToolButton;
    btnLargeIcons: TToolButton;
    btnSmallIcons: TToolButton;
    btnList: TToolButton;
    btnReport: TToolButton;
    cbPath: TComboBox;
    ToolButton3: TToolButton;
    PopupMenu1: TPopupMenu;
    btnBack: TToolButton;
    <--------> procedure
FormCreate(Sender: TObject);
    <--------> procedure
ListViewData(Sender: TObject;
Item: TListItem);
    <--------> procedure
btnBrowseClick(Sender: TObject);
    <--------> procedure
cbPathKeyDown(Sender: TObject;
var Key: Word;
      Shift: TShiftState);
    <--------> procedure
cbPathClick(Sender: TObject);
    <--------> procedure
btnLargeIconsClick(Sender:
TObject);
    <--------> procedure
ListViewDblClick(Sender: TObject);
    <--------> procedure
ListViewDataHint(Sender: TObject;
StartIndex,
      EndIndex: Integer);
    <--------> procedure
ListViewKeyDown(Sender:
TObject; var Key: Word;
      Shift: TShiftState);
    <--------> procedure
ListViewDataFind(Sender: TObject;
Find: TItemFind;
      const FindString: String; const
FindPosition: TPoint;
```

```
uses ShellAPI, ActiveX, ComObj, CommCtrl, FileCtrl;

//PIDL MANIPULATION

<-------> procedure DisposePIDL(ID: PItemIDList);
var
  Malloc: IMalloc;
begin
  if ID = nil then Exit;
  OLECheck(SHGetMalloc(Malloc));
  Malloc.Free(ID);
end;

function CopyITEMID(Malloc: IMalloc; ID: PItemIDList): PItemIDList;
begin
  Result := Malloc.Alloc(ID^.mkid.cb + SizeOf(ID^.mkid.cb));
  CopyMemory(Result, ID, ID^.mkid.cb + SizeOf(ID^.mkid.cb));
end;

function NextPIDL(IDList: PItemIDList): PItemIDList;
begin
  Result := IDList;
  Inc(PChar(Result), IDList^.mkid.cb);
end;

function GetPIDLSize(IDList: PItemIDList): Integer;
begin
  Result := 0;
  if Assigned(IDList) then
  begin
    Result := SizeOf(IDList^.mkid.cb);
    while IDList^.mkid.cb <> 0 do
    begin
      Result := Result + IDList^.mkid.cb;
      IDList := NextPIDL(IDList);
    end;
```

```
      FindData: Pointer; StartIndex: Integer; Direction: TSearchDirection;
      Wrap: Boolean; var Index: Integer);
    <-------> procedure ListViewCustomDrawItem(Sender: TCustomListView;
      Item: TListItem; State: TCustomDrawState; var DefaultDraw: Boolean);
    <-------> procedure ListViewCustomDrawSubItem(Sender: TCustomListView;
      Item: TListItem; SubItem: Integer; State: TCustomDrawState;
      var DefaultDraw: Boolean);
    <-------> procedure btnBackClick(Sender: TObject);
    <-------> procedure Form1Close(Sender: TObject; var Action: TCloseAction);
  private
    FPIDL: PItemIDList;
    FIDList: TList;
    FIShellFolder,
    FIDesktopFolder: IShellFolder;
    FPath: string;
    <-------> procedure SetPath(const Value: string); overload;
    <-------> procedure SetPath(ID: PItemIDList); overload;
    <-------> procedure PopulateIDList(ShellFolder: IShellFolder);
    <-------> procedure ClearIDList;
    <-------> procedure CheckShellItems(StartIndex, EndIndex: Integer);
    function  ShellItem(Index: Integer): PShellItem;
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}
```

```
  if Assigned(Result) then
    CopyMemory(Result, IDList,
Size);
end;

function ConcatPIDLs(IDList1,
IDList2: PItemIDList):
PItemIDList;
var
  cb1, cb2: Integer;
begin
  if Assigned(IDList1) then
    cb1 := GetPIDLSize(IDList1) -
SizeOf(IDList1^.mkid.cb)
  else
    cb1 := 0;

  cb2 := GetPIDLSize(IDList2);

  Result := CreatePIDL(cb1 + cb2);
  if Assigned(Result) then
  begin
    if Assigned(IDList1) then
      CopyMemory(Result, IDList1,
cb1);
    CopyMemory(PChar(Result) +
cb1, IDList2, cb2);
  end;
end;

//SHELL FOLDER ITEM INFO

function
GetDisplayName(ShellFolder:
IShellFolder; PIDL: PItemIDList;
                        ForParsing: Boolean):
string;
var
  StrRet: TStrRet;
  P: PChar;
  Flags: Integer;
begin
  Result := '';
  if ForParsing then
    Flags := SHGDN_FORPARSING
  else
    Flags := SHGDN_NORMAL;

  end;
end;


<-------> procedure
StripLastID(IDList: PItemIDList);
var
  MarkerID: PItemIDList;
begin
  MarkerID := IDList;
  if Assigned(IDList) then
  begin
     while IDList.mkid.cb <> 0 do
    begin
      MarkerID := IDList;
      IDList := NextPIDL(IDList);
    end;
    MarkerID.mkid.cb := 0;
  end;
end;

function CreatePIDL(Size: Integer):
PItemIDList;
var
  Malloc: IMalloc;
  HR: HResult;
begin
  Result := nil;

  HR := SHGetMalloc(Malloc);
  if Failed(HR) then
    Exit;

  try
    Result := Malloc.Alloc(Size);
    if Assigned(Result) then
      FillChar(Result^, Size, 0);
  finally
  end;
end;

function CopyPIDL(IDList:
PItemIDList): PItemIDList;
var
  Size: Integer;
begin
  Size := GetPIDLSize(IDList);
  Result := CreatePIDL(Size);
```

```
var
  Flags: UINT;
begin
  Flags := SFGAO_FOLDER;
  ShellFolder.GetAttributesOf(1, ID,
Flags);
  Result := SFGAO_FOLDER and
Flags <> 0;
end;


function ListSortFunc(Item1, Item2:
Pointer): Integer;
begin
  Result :=
SmallInt(Form1.FIShellFolder.Com
pareIDs(
            0,
            PShellItem(Item1).ID,
            PShellItem(Item2).ID
        ));
end;

{TForm1}

//GENERAL FORM METHODS

<-------> procedure
TForm1.FormCreate(Sender:
TObject);
var
  FileInfo: TSHFileInfo;
  ImageListHandle: THandle;
  NewPIDL: PItemIDList;
begin

OLECheck(SHGetDesktopFolder(FI
DesktopFolder));
  FIShellFolder := FIDesktopFolder;
  FIDList := TList.Create;
  ImageListHandle :=
SHGetFileInfo('C:\',
                0,
                FileInfo,
                SizeOf(FileInfo),

SHGFI_SYSICONINDEX or
SHGFI_SMALLICON);
  SendMessage(ListView.Handle,
```

```

ShellFolder.GetDisplayNameOf(PID
L, Flags, StrRet);
  case StrRet.uType of
    STRRET_CSTR:
      SetString(Result, StrRet.cStr,
lStrLen(StrRet.cStr));
    STRRET_OFFSET:
      begin
        P :=
@PIDL.mkid.abID[StrRet.uOffset -
SizeOf(PIDL.mkid.cb)];
        SetString(Result, P,
PIDL.mkid.cb - StrRet.uOffset);
      end;
    STRRET_WSTR:
      Result := StrRet.pOleStr;
  end;
end;

function GetShellImage(PIDL:
PItemIDList; Large, Open:
Boolean): Integer;
var
  FileInfo: TSHFileInfo;
  Flags: Integer;
begin
  FillChar(FileInfo, SizeOf(FileInfo),
#0);
  Flags := SHGFI_PIDL or
SHGFI_SYSICONINDEX or
SHGFI_ICON;
  if Open then Flags := Flags or
SHGFI_OPENICON;
  if Large then Flags := Flags or
SHGFI_LARGEICON
  else Flags := Flags or
SHGFI_SMALLICON;
  SHGetFileInfo(PChar(PIDL),
              0,
              FileInfo,
              SizeOf(FileInfo),
              Flags);
  Result := FileInfo.iIcon;
end;

function IsFolder(ShellFolder:
IShellFolder; ID: PItemIDList):
Boolean;
```

```
cbPath.Text[Length(cbPath.Text)] =
':' then
      cbPath.Text := cbPath.Text + '\';
    SetPath(cbPath.Text);
    Key := 0;
  end;
end;

<-------> procedure
TForm1.cbPathClick(Sender:
TObject);
var
  I: Integer;
begin
  I :=
cbPath.Items.IndexOf(cbPath.Text);
  if I >= 0 then

SetPath(PItemIDList(cbPath.Items.
Objects[I]))
  else
    SetPath(cbPath.Text);
end;

<-------> procedure
TForm1.btnLargeIconsClick(Sender
: TObject);
begin
  ListView.ViewStyle :=
TViewStyle((Sender as
TComponent).Tag);
end;

<-------> procedure
TForm1.ListViewDblClick(Sender:
TObject);
var
  RootPIDL,
  ID: PItemIDList;
begin
  if ListView.Selected <> nil then
  begin
    ID :=
ShellItem(ListView.Selected.Index).I
D;
    if not IsFolder(FIShellFolder, ID)
then Exit;
    RootPIDL :=
ConcatPIDLs(FPIDL, ID);
```

```
LVM_SETIMAGELIST,
LVSIL_SMALL, ImageListHandle);

  ImageListHandle :=
SHGetFileInfo('C:\',
                    0,
                    FileInfo,
                    SizeOf(FileInfo),

SHGFI_SYSICONINDEX or
SHGFI_LARGEICON);

  SendMessage(ListView.Handle,
LVM_SETIMAGELIST,
LVSIL_NORMAL,
ImageListHandle);
  OLECheck(
    SHGetSpecialFolderLocation(
      Application.Handle,
      CSIDL_DRIVES,
      NewPIDL)
  );
  SetPath(NewPIDL);
  ActiveControl := cbPath;
  cbPath.SelStart := 0;
  cbPath.SelLength :=
Length(cbPath.Text);
end;

<-------> procedure
TForm1.btnBrowseClick(Sender:
TObject);
var
  S: string;
begin
  S := '';
  if SelectDirectory('Select
Directory', '', S) then
    SetPath(S);
end;

<-------> procedure
TForm1.cbPathKeyDown(Sender:
TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key = VK_RETURN then
  begin
    if
```

```
  EnumList: IEnumIDList;
  NumIDs: LongWord;
  SaveCursor: TCursor;
  ShellItem: PShellItem;
begin
  SaveCursor := Screen.Cursor;
  try
    Screen.Cursor := crHourglass;
    OleCheck(
      ShellFolder.EnumObjects(
        Application.Handle,
        Flags,
        EnumList)
    );

    FIShellFolder := ShellFolder;
    ClearIDList;
    while EnumList.Next(1, ID,
NumIDs) = S_OK do
    begin
      ShellItem := New(PShellItem);
      ShellItem.ID := ID;
      ShellItem.DisplayName :=
GetDisplayName(FIShellFolder, ID,
False);
      ShellItem.Empty := True;
      FIDList.Add(ShellItem);
    end;

    FIDList.Sort(ListSortFunc);

    //We need to tell the ListView how
many items it has.
    ListView.Items.Count :=
FIDList.Count;

    ListView.Repaint;
  finally
    Screen.Cursor := SaveCursor;
  end;
end;

<-------> procedure
TForm1.SetPath(const Value:
string);
var
  P: PWideChar;
  NewPIDL: PItemIDList;
    SetPath(RootPIDL);
  end;
end;

function TForm1.ShellItem(Index:
Integer): PShellItem;
begin
  Result :=
PShellItem(FIDList[Index]);
end;

<-------> procedure
TForm1.ListViewKeyDown(Sender:
TObject; var Key: Word;
  Shift: TShiftState);
begin
  case Key of
    VK_RETURN:
      ListViewDblClick(Sender);
    VK_BACK:
      btnBackClick(Sender);
  end;
end;

//SHELL-RELATED ROUTINES.

<-------> procedure
TForm1.ClearIDList;
var
  I: Integer;
begin
  for I := 0 to FIDList.Count-1 do
  begin
    DisposePIDL(ShellItem(I).ID);
    Dispose(ShellItem(I));
  end;
  FIDList.Clear;
end;

<-------> procedure
TForm1.PopulateIDList(ShellFolder
: IShellFolder);
const
  Flags = SHCONTF_FOLDERS or
SHCONTF_NONFOLDERS or
SHCONTF_INCLUDEHIDDEN;
var
  ID: PItemIDList;
```

```
    cbPath.Text := cbPath.Items[0];
  end
  else begin
    cbPath.ItemIndex := Index;
    cbPath.Text :=
cbPath.Items[cbPath.ItemIndex];
  end;

  if ListView.Items.Count > 0 then
  begin
    ListView.Selected :=
ListView.Items[0];
    ListView.Selected.Focused :=
True;

ListView.Selected.MakeVisible(False
);
  end;
 finally
  ListView.Items.EndUpdate;
 end;
end;

//ROUTINES FOR MANAGING
VIRTUAL DATA

<-------> procedure
TForm1.CheckShellItems(StartInde
x, EndIndex: Integer);

 function ValidFileTime(FileTime:
TFileTime): Boolean;
 begin
   Result :=
(FileTime.dwLowDateTime <> 0) or
(FileTime.dwHighDateTime <> 0);
 end;

var
 FileData: TWin32FindData;
 FileInfo: TSHFileInfo;
 SysTime: TSystemTime;
 I: Integer;
 LocalFileTime: TFILETIME;
begin
 //Here all the data that wasn't
initialized in PopulateIDList is
 //filled in.
 for I := StartIndex to EndIndex do
```

```
 Flags,
 NumChars: LongWord;
begin
 NumChars := Length(Value);
 Flags := 0;
 P := StringToOleStr(Value);

 OLECheck(

FIDesktopFolder.ParseDisplayName
(
    Application.Handle,
    nil,
    P,
    NumChars,
    NewPIDL,
    Flags)
  );
 SetPath(NewPIDL);
end;

<-------> procedure
TForm1.SetPath(ID: PItemIDList);
var
 Index: Integer;
 NewShellFolder: IShellFolder;
begin
  OLECheck(
   FIDesktopFolder.BindToObject(
       ID,
       nil,
       IID_IShellFolder,
       Pointer(NewShellFolder))
  );

 ListView.Items.BeginUpdate;
 try
  PopulateIDList(NewShellFolder);
  FPIDL := ID;
  FPath :=
GetDisplayName(FIDesktopFolder,
FPIDL, True);
  Index :=
cbPath.Items.IndexOf(FPath);
  if (Index < 0) then
  begin
    cbPath.Items.InsertObject(0,
FPath, Pointer(FPIDL));
```

```
me, SysTime) then
      try
        ModDate :=
DateTimeToStr(SystemTimeToDate
Time(SysTime))
      except
        on EConvertError do
ModDate := '';
      end
      else
        ModDate := '';

    //Attributes
    Attributes :=
FileData.dwFileAttributes;

    //Flag this record as complete.
    Empty := False;
  end;
 end;
end;

<-------> procedure
TForm1.ListViewDataHint(Sender:
TObject; StartIndex,
  EndIndex: Integer);
begin
  //OnDataHint is called before
OnData. This gives you a chance to
  //initialize only the data structures
that need to be drawn.
  //You should keep track of which
items have been initialized so no
  //extra work is done.
  if (StartIndex > FIDList.Count) or
(EndIndex > FIDList.Count) then
Exit;
  CheckShellItems(StartIndex,
EndIndex);
end;

<-------> procedure
TForm1.ListViewData(Sender:
TObject; Item: TListItem);
var
  Attrs: string;
begin
  //OnData gets called once for each
item for which the ListView needs
```

```
  begin
   if ShellItem(I)^.Empty then
   with ShellItem(I)^ do
   begin
     FullID := ConcatPIDLs(FPIDL,
ID);
     ImageIndex :=
GetShellImage(FullID,
ListView.ViewStyle = vsIcon, False);

     //File Type
     SHGetFileInfo(
       PChar(FullID),
       0,
       FileInfo,
       SizeOf(FileInfo),
       SHGFI_TYPENAME or
SHGFI_PIDL
     );
     TypeName :=
FileInfo.szTypeName;

     //Get File info from Windows
     FillChar(FileData,
SizeOf(FileData), #0);
     SHGetDataFromIDList(
       FIShellFolder,
       ID,
       SHGDFIL_FINDDATA,
       @FileData,
       SizeOf(FileData)
     );

     //File Size, in KB
     Size := (FileData.nFileSizeLow +
1023 ) div 1024;
     if Size = 0 then Size := 1;

     //Modified Date
     FillChar(LocalFileTime,
SizeOf(TFileTime), #0);
     with FileData do
       if
ValidFileTime(ftLastWriteTime)
       and
FileTimeToLocalFileTime(ftLastWr
iteTime, LocalFileTime)
       and
FileTimeToSystemTime(LocalFileTi
```

```
FindPosition: TPoint; FindData: Pointer;
  StartIndex: Integer; Direction: TSearchDirection; Wrap: Boolean;
  var Index: Integer);
//OnDataFind gets called in response to calls to FindCaption, FindData,
//GetNearestItem, etc. It also gets called for each keystroke sent to the
//ListView (for incremental searching)
var
  I: Integer;
  Found: Boolean;
begin
  I := StartIndex;
  if (Find = ifExactString) or (Find = ifPartialString) then
  begin
    repeat
      if (I = FIDList.Count-1) then
        if Wrap then I := 0 else Exit;
      Found := Pos(UpperCase(FindString), UpperCase(ShellItem(I)^.DisplayName)) = 1;
      Inc(I);
    until Found or (I = StartIndex);
    if Found then Index := I-1;
  end;
end;

<-------> procedure TForm1.ListViewCustomDrawItem(Sender: TCustomListView;
  Item: TListItem; State: TCustomDrawState; var DefaultDraw: Boolean);
var
  Attrs: Integer;
begin
  if Item = nil then Exit;
  Attrs := ShellItem(Item.Index).Attributes;
  if Bool(Attrs and FILE_ATTRIBUTE_READONLY) then
    ListView.Canvas.Font.Color := clGrayText;
```

```
  //data. If the ListView is in Report View, be sure to add the subitems.
  //Item is a "dummy" item whose only valid data is it's index which
  //is used to index into the underlying data.
  if (Item.Index > FIDList.Count) then Exit;
  with ShellItem(Item.Index)^ do
  begin
    Item.Caption := DisplayName;
    Item.ImageIndex := ImageIndex;

    if ListView.ViewStyle <> vsReport then Exit;

    if not IsFolder(FIShellFolder, ID) then
      Item.SubItems.Add(Format('%dKB ', [Size]))
    else
      Item.SubItems.Add('');
    Item.SubItems.Add(TypeName);
    try
      Item.SubItems.Add(ModDate);
    except
    end;

    if Bool(Attributes and FILE_ATTRIBUTE_READONLY) then Attrs := Attrs + 'R';
    if Bool(Attributes and FILE_ATTRIBUTE_HIDDEN) then Attrs := Attrs + 'H';
    if Bool(Attributes and FILE_ATTRIBUTE_SYSTEM) then Attrs := Attrs + 'S';
    if Bool(Attributes and FILE_ATTRIBUTE_ARCHIVE) then Attrs := Attrs + 'A';
  end;
  Item.SubItems.Add(Attrs);
end;

<-------> procedure TForm1.ListViewDataFind(Sender: TObject; Find: TItemFind;
  const FindString: String; const
```

<-------> object s14

الملف النصي لبرنامج

Virtual Listview)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<-------> object Form1: TForm1
  Left = 174
  Top = 113
  Width = 569
  Height = 448
  Caption = 'Virtual List View Demo'
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  Visible = True
  OnClose = Form1Close
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object ListView:
TListView
    Left = 0
    Top = 56
    Width = 561
    Height = 358
    Align = alClient
    Columns = <
      item
        Caption = 'Name'
        Width = 175
      end
      item
        Alignment = taRightJustify
        Caption = 'Size'
        Width = 75
      end
      item
        Caption = 'Type'
        Width = 150
      end
      item
```

```
  if Bool(Attrs and
FILE_ATTRIBUTE_HIDDEN) then
    ListView.Canvas.Font.Style :=
      ListView.Canvas.Font.Style +
[fsStrikeOut];
  if Bool(Attrs and
FILE_ATTRIBUTE_SYSTEM) then
    Listview.Canvas.Font.Color :=
clHighlight;
end;

<-------> procedure
TForm1.ListViewCustomDrawSubIt
em(Sender: TCustomListView;
  Item: TListItem; SubItem: Integer;
State: TCustomDrawState;
  var DefaultDraw: Boolean);
begin
  if SubItem = 0 then Exit;
  ListView.Canvas.Font.Color :=
GetSysColor(COLOR_WINDOWT
EXT);
  //workaround for Win98 bug.
end;

<-------> procedure
TForm1.btnBackClick(Sender:
TObject);
var
  Temp: PItemIDList;
begin
  Temp := CopyPIDL(FPIDL);
  if Assigned(Temp) then
    StripLastID(Temp);
  if Temp.mkid.cb <> 0 then
    SetPath(Temp)
  else
    Beep;
end;

<-------> procedure
TForm1.Form1Close(Sender:
TObject; var Action: TCloseAction);
begin
  ClearIDList;
  FIDList.Free;
end;

end.
```

AutoSize = True
ButtonHeight = 27
ButtonWidth = 28
Caption = 'ToolBar2'
Flat = True
Images = ToolbarImages
TabOrder = 0
<-------> object btnBrowse: TToolButton
Left = 0
Top = 0
Caption = 'btnBrowse'
ImageIndex = 0
OnClick = btnBrowseClick
end
<-------> object btnBack: TToolButton
Left = 28
Top = 0
Caption = 'btnBack'
ImageIndex = 5
OnClick = btnBackClick
end
<-------> object ToolButton3: TToolButton
Left = 56
Top = 0
Width = 8
Caption = 'ToolButton3'
ImageIndex = 5
Style = tbsSeparator
end
<-------> object btnLargeIcons: TToolButton
Left = 64
Top = 0
Caption = 'btnLargeIcons'
Grouped = True
ImageIndex = 1
Style = tbsCheck
OnClick = btnLargeIconsClick
end
<-------> object btnSmallIcons: TToolButton
Tag = 1
Left = 92
Top = 0
Caption = 'btnSmallIcons'

Caption = 'Modified'
Width = 100
end
item
Alignment = taRightJustify
Caption = 'Attributes'
Width = 75
end>
OwnerData = True
ReadOnly = True
TabOrder = 0
ViewStyle = vsReport
OnCustomDrawItem = ListViewCustomDrawItem
OnCustomDrawSubItem = ListViewCustomDrawSubItem
OnData = ListViewData
OnDataFind = ListViewDataFind
OnDataHint = ListViewDataHint
OnDblClick = ListViewDblClick
OnKeyDown = ListViewKeyDown
end
<-------> object CoolBar1: TCoolBar
Left = 0
Top = 0
Width = 561
Height = 56
AutoSize = True
Bands = <
item
Control = ToolBar2
ImageIndex = -1
MinHeight = 29
Width = 557
end
item
Control = cbPath
ImageIndex = -1
MinHeight = 21
Width = 557
end>
<-------> object ToolBar2: TToolBar
Left = 9
Top = 0
Width = 544
Height = 29

```
        Grouped = True
        ImageIndex = 2
        Style = tbsCheck
        OnClick = btnLargeIconsClick
      end
      <-------> object btnList:
TToolButton
        Tag = 2
        Left = 120
        Top = 0
        Caption = 'btnList'
        Grouped = True
        ImageIndex = 3
        Style = tbsCheck
        OnClick = btnLargeIconsClick
      end
      <-------> object btnReport:
TToolButton
        Tag = 3
        Left = 148
        Top = 0
        Caption = 'btnReport'
        Grouped = True
        ImageIndex = 4
        Style = tbsCheck
        OnClick = btnLargeIconsClick
      end
    end
    <-------> object cbPath:
TComboBox
      Left = 9
      Top = 31
      Width = 544
      Height = 21
      ItemHeight = 13
      TabOrder = 1
      OnClick = cbPathClick
      OnKeyDown = cbPathKeyDown
    end
  end
  <-------> object ToolbarImages:
TImageList
    Height = 21
    Width = 21
    Left = 72
    Top = 128
    Bitmap = {

0000000000000000000000000000}
  end
  <-------> object PopupMenu1:
TPopupMenu
    Left = 160
    Top = 128
  end
end
```

```
Application.CreateForm(TFileAttrForm, FileAttrForm);
Application.CreateForm(TChangeDlg, ChangeDlg);
  Application.Run;
end.

\\\\\\\\\\\\\\\\

unit FChngDlg;

interface

uses Windows, SysUtils, Classes,
Graphics, Forms, Controls, StdCtrls,
  Buttons, ExtCtrls;

type
  TChangeDlg = class(TForm)
    OKBtn: TButton;
    CancelBtn: TButton;
    Bevel1: TBevel;
    CurrentDir: TLabel;
    Label1: TLabel;
    Label2: TLabel;
    ToFileName: TEdit;
    FromFileName: TEdit;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  ChangeDlg: TChangeDlg;

implementation

{$R *.dfm}

end.

\\\\\\\\\\\\\\\\

unit FAttrDlg;
```

الملف التنفيذي لبرنامج

34 procedures

Filmanex)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
program FilManEx;

uses
  Forms,
  FMXWin in 'FMXWin.pas' {FMForm},
  FmxUtils in 'Fmxutils.pas',
  FAttrDlg in 'FAttrDlg.pas' {FileAttrForm},
  FChngDlg in 'FChngDlg.pas' {ChangeDlg};

{$R *.RES}

begin
  Application.Initialize;
Application.CreateForm(TFMForm, FMForm);
```

StatusBar: TPanel;
DirectoryPanel: TPanel;
FilePanel: TPanel;
DriveTabSet: TTabSet;
DirectoryOutline: TDirectoryOutline;
FileList: TFileListBox;
MainMenu1: TMainMenu;
File1: TMenuItem;
Open1: TMenuItem;
Move1: TMenuItem;
Copy1: TMenuItem;
Delete1: TMenuItem;
Rename1: TMenuItem;
Properties1: TMenuItem;
N1: TMenuItem;
Exit1: TMenuItem;
Floppy: TImage;
Fixed: TImage;
Network: TImage;
CDRom: TImage;
RamDisk: TImage;
<-------> procedure Exit1Click(Sender: TObject);
<-------> procedure FormCreate(Sender: TObject);
<-------> procedure DirectoryOutlineChange(Sender: TObject);
<-------> procedure FileListChange(Sender: TObject);
<-------> procedure DriveTabSetMeasureTab(Sender: TObject; Index: Integer;
var TabWidth: Integer);
<-------> procedure DriveTabSetDrawTab(Sender: TObject; TabCanvas: TCanvas;
R: TRect; Index: Integer; Selected: Boolean);
<-------> procedure File1Click(Sender: TObject);
<-------> procedure Delete1Click(Sender: TObject);
<-------> procedure Properties1Click(Sender: TObject);
<-------> procedure FileChange(Sender: TObject);
<-------> procedure

interface
uses Windows, SysUtils, Classes, Graphics, Forms, Controls, StdCtrls,
Buttons, ExtCtrls;
type
TFileAttrForm = class(TForm)
OKBtn: TButton;
CancelBtn: TButton;
Bevel1: TBevel;
FileDirName: TLabel;
FilePathName: TLabel;
ChangeDate: TLabel;
GroupBox1: TGroupBox;
ReadOnly: TCheckBox;
Archive: TCheckBox;
System: TCheckBox;
Hidden: TCheckBox;
private
{ Private declarations }
public
{ Public declarations }
end;
var
FileAttrForm: TFileAttrForm;
implementation
{$R *.dfm}
end.
\\\\\\\\\\\\\\\\\\\\\\
unit FMXWin;
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, FileCtrl, Grids, Outline, DirOutln, Tabs, ExtCtrls, Menus;
type
TFMForm = class(TForm)

```
var
  Drive: Char;
  AddedIndex: Integer;
begin
  for Drive := 'a' to 'z' do
  begin
    case GetDriveType(PChar(Drive
+ ':\')) of
      DRIVE_REMOVABLE:
        AddedIndex :=
DriveTabSet.Tabs.AddObject(Drive,
Floppy.Picture.Graphic);
      DRIVE_FIXED:
        AddedIndex :=
DriveTabSet.Tabs.AddObject(Drive,
Fixed.Picture.Graphic);
      DRIVE_CDROM:
        AddedIndex :=
DriveTabSet.Tabs.AddObject(Drive,
CDRom.Picture.Graphic);
      DRIVE_RAMDISK:
        AddedIndex :=
DriveTabSet.Tabs.AddObject(Drive,
RamDisk.Picture.Graphic);
      DRIVE_REMOTE:
        AddedIndex :=
DriveTabSet.Tabs.AddObject(Drive,
Network.Picture.Graphic);
    else
      AddedIndex := 0;
    end;
    if UpCase(Drive) = FileList.Drive
then
      DriveTabSet.TabIndex :=
AddedIndex;
  end;
end;

<-------> procedure
TFMForm.DriveTabSetChange(Sen
der: TObject; NewTab: Integer;
  var AllowChange: Boolean);
begin
  if not (csDesigning in
ComponentState) then
  begin
    AllowChange := True;
    try
      with DriveTabSet do
```

```
Open1Click(Sender: TObject);
    <-------> procedure
FileListMouseDown(Sender:
TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y:
Integer);
    <-------> procedure
DirectoryOutlineDragOver(Sender,
Source: TObject; X,
      Y: Integer; State: TDragState;
var Accept: Boolean);
    <-------> procedure
DirectoryOutlineDragDrop(Sender,
Source: TObject; X,
      Y: Integer);
    <-------> procedure
FileListEndDrag(Sender, Target:
TObject; X, Y: Integer);
    <-------> procedure
DriveTabSetChange(Sender:
TObject; NewTab: Integer;
      var AllowChange: Boolean);
  private
    <-------> procedure
ConfirmChange(const ACaption,
FromFile, ToFile: string);
  public
    { Public declarations }
  end;

var
  FMForm: TFMForm;

implementation

uses FmxUtils, FAttrDlg, FChngDlg;

{$R *.dfm}

<-------> procedure
TFMForm.Exit1Click(Sender:
TObject);
begin
  Close;
end;

<-------> procedure
TFMForm.FormCreate(Sender:
TObject);
```

```
  var TabWidth: Integer);
var
  BitmapWidth: Integer;
begin
  BitmapWidth :=
TBitmap(DriveTabSet.Tabs.Objects
[Index]).Width;
  Inc(TabWidth, 2 + BitmapWidth);
end;

<-------> procedure
TFMForm.DriveTabSetDrawTab(Se
nder: TObject; TabCanvas:
TCanvas;
  R: TRect; Index: Integer; Selected:
Boolean);
var
  Bitmap: TBitmap;
begin
  Bitmap :=
TBitmap(DriveTabSet.Tabs.Objects
[Index]);
  with TabCanvas do
  begin
    Draw(R.Left, R.Top + 4, Bitmap);
    TextOut(R.Left + 2 +
Bitmap.Width, R.Top + 2,
DriveTabSet.Tabs[Index]);
  end;
end;

<-------> procedure
TFMForm.File1Click(Sender:
TObject);
var
  FileSelected: Boolean;
begin
  FileSelected := FileList.ItemIndex
>= 0;
  Open1.Enabled := FileSelected;
  Delete1.Enabled := FileSelected;
  Copy1.Enabled := FileSelected;
  Move1.Enabled := FileSelected;
  Rename1.Enabled := FileSelected;
  Properties1.Enabled :=
FileSelected;
end;

<-------> procedure
```

```
      DirectoryOutline.Drive :=
Tabs[NewTab][1];
    except
      on EInOutError do
      begin
        AllowChange := False;
        with DriveTabSet do
          DirectoryOutline.Drive :=
Tabs[TabIndex][1];
        raise;
      end;
    end;
  end;
end;

<-------> procedure
TFMForm.DirectoryOutlineChange(
Sender: TObject);
begin
  FileList.Directory :=
DirectoryOutline.Directory;
  DirectoryPanel.Caption :=
DirectoryOutline.Directory;
end;

<-------> procedure
TFMForm.FileListChange(Sender:
TObject);
var
  TheFileName: string;
begin
  with FileList do
  begin
    if ItemIndex >= 0 then
    begin
      TheFileName :=
Items[ItemIndex];
      FilePanel.Caption :=
Format('%s, %d bytes',
[TheFileName,
GetFileSize(TheFileName)]);
    end
    else FilePanel.Caption := '';
  end;
end;

<-------> procedure
TFMForm.DriveTabSetMeasureTab
(Sender: TObject; Index: Integer;
```

```
    if System.Checked then NewAttributes := NewAttributes or faSysFile
    else NewAttributes := NewAttributes and not faSysFile;
    if Hidden.Checked then NewAttributes := NewAttributes or faHidden
    else NewAttributes := NewAttributes and not faHidden;
    if NewAttributes <> Attributes then
      FileSetAttr(FileDirName.Caption, NewAttributes);
   end;
  end;
end;

<-------> procedure TFMForm.ConfirmChange(const ACaption, FromFile, ToFile: string);
begin
  if MessageDlg(Format('%s %s to %s?', [ACaption, FromFile, ToFile]),
   mtConfirmation, [mbYes, mbNo], 0) = mrYes then
  begin
   if ACaption = 'Move' then
    MoveFile(FromFile, ToFile)
   else if ACaption = 'Copy' then
    CopyFile(FromFile, ToFile)
   else if ACaption = 'Rename' then
    RenameFile(FromFile, ToFile);
   FileList.Update;
  end;
end;

<-------> procedure TFMForm.FileChange(Sender: TObject);
begin
  with ChangeDlg do
  begin
   if Sender = Move1 then Caption := 'Move'
   else if Sender = Copy1 then Caption := 'Copy'
TFMForm.Delete1Click(Sender: TObject);
begin
  with FileList do
   if MessageDlg('Delete ' + FileName + '?', mtConfirmation,
    [mbYes, mbNo], 0) = mrYes then
    if DeleteFile(FileName) then Update;
end;

<-------> procedure TFMForm.Properties1Click(Sender: TObject);
var
  Attributes, NewAttributes: Word;
begin
  with FileAttrForm do
  begin
   FileDirName.Caption := FileList.Items[FileList.ItemIndex];
   FilePathName.Caption := FileList.Directory;
   ChangeDate.Caption := DateTimeToStr(FileDateTime(FileList.FileName));
   Attributes := FileGetAttr(FileDirName.Caption);
   ReadOnly.Checked := (Attributes and faReadOnly) = faReadOnly;
   Archive.Checked := (Attributes and faArchive) = faArchive;
   System.Checked := (Attributes and faSysFile) = faSysFile;
   Hidden.Checked := (Attributes and faHidden) = faHidden;
   if ShowModal <> mrCancel then
   begin
    NewAttributes := Attributes;
    if ReadOnly.Checked then NewAttributes := NewAttributes or faReadOnly
    else NewAttributes := NewAttributes and not faReadOnly;
    if Archive.Checked then NewAttributes := NewAttributes or faArchive
    else NewAttributes := NewAttributes and not faArchive;
```

```
  Y: Integer; State: TDragState; var
Accept: Boolean);
begin
  Accept := (Source is TFileListBox)
and (DirectoryOutline.GetItem(X,
Y) > 0);
end;

<-------> procedure
TFMForm.DirectoryOutlineDragDr
op(Sender, Source: TObject; X,
  Y: Integer);
begin
  if Source is TFileListBox then
    with DirectoryOutline do
      ConfirmChange('Move',
FileList.FileName, Items[GetItem(X,
Y)].FullPath);
end;

<-------> procedure
TFMForm.FileListEndDrag(Sender,
Target: TObject; X, Y: Integer);
begin
  if Target <> nil then
FileList.Update;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

<-------> object s43

الملف النصي لبرنامج

Filmanex)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<-------> object FMForm: TFMForm
  Left = -4
  Top = -4
  Width = 1032
  Height = 746
  Caption = 'File manager example'
```

```
    else if Sender = Rename1 then
Caption := 'Rename'
    else Exit;
    CurrentDir.Caption :=
DirectoryOutline.Directory;
    FromFileName.Text :=
FileList.FileName;
    ToFileName.Text := '';
    if (ShowModal <> mrCancel) and
(ToFileName.Text <> '') then
      ConfirmChange(Caption,
FromFileName.Text,
ToFileName.Text);
  end;
end;

<-------> procedure
TFMForm.Open1Click(Sender:
TObject);
begin
  with FileList do
  begin
    if HasAttr(FileName, faDirectory)
then
      DirectoryOutline.Directory :=
FileName
    else ExecuteFile(FileName, '',
Directory, SW_SHOW);
  end;
end;

<-------> procedure
TFMForm.FileListMouseDown(Sen
der: TObject; Button:
TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  if Button = mbLeft then
  with Sender as TFileListBox do
  begin
    if ItemAtPos(Point(X, Y), True)
>= 0 then
      BeginDrag(False);
  end;
end;

<-------> procedure
TFMForm.DirectoryOutlineDragOv
er(Sender, Source: TObject; X,
```

```
    Visible = False
  end
  <-------> object Fixed: TImage
    Left = 48
    Top = 16
    Width = 16
    Height = 9
    AutoSize = True
    Picture.Data = {
    Visible = False
  end
  <-------> object Network: TImage
    Left = 72
    Top = 16
    Width = 16
    Height = 9
    AutoSize = True
    Picture.Data = {
    Visible = False
  end
  <-------> object CDRom: TImage
    Left = 98
    Top = 16
    Width = 16
    Height = 9
    AutoSize = True
    Picture.Data = {

    Visible = False
  end
  <-------> object RamDisk:
TImage
    Left = 118
    Top = 16
    Width = 16
    Height = 9
    AutoSize = True
    Picture.Data = {
                              }
    Visible = False
  end
 end
end
<-------> object DriveTabSet:
TTabSet
 Left = 0
 Top = 631
 Width = 1024
```

```
 Color = clBtnFace
 Font.Charset =
DEFAULT_CHARSET
 Font.Color = clWindowText
 Font.Height = -11
 Font.Name = 'MS Sans Serif'
 Font.Style = []
 Menu = MainMenu1
 OldCreateOrder = True
 Position = poDefault
 OnCreate = FormCreate
 PixelsPerInch = 96
 TextHeight = 13
 <-------> object StatusBar: TPanel
  Left = 0
  Top = 652
  Width = 1024
  Height = 40
  Align = alBottom
  BevelOuter = bvNone
  TabOrder = 0
  <-------> object DirectoryPanel:
TPanel
   Left = 0
   Top = 0
   Width = 185
   Height = 40
   Align = alLeft
   BevelInner = bvLowered
   BevelWidth = 2
   TabOrder = 0
  end
  <-------> object FilePanel: TPanel
   Left = 185
   Top = 0
   Width = 839
   Height = 40
   Align = alClient
   BevelInner = bvLowered
   BevelWidth = 2
   TabOrder = 1
   <-------> object Floppy: TImage
    Left = 24
    Top = 16
    Width = 16
    Height = 9
    AutoSize = True
    Picture.Data = {
```

```
  OnMouseDown = FileListMouseDown
 end
 <-------> object MainMenu1: TMainMenu
  Left = 337
  Top = 251
  <-------> object File1: TMenuItem
   Caption = '&File'
   OnClick = File1Click
   <-------> object Open1: TMenuItem
    Caption = '&Open'
    ShortCut = 13
    OnClick = Open1Click
   end
   <-------> object Move1: TMenuItem
    Caption = '&Move...'
    ShortCut = 118
    OnClick = FileChange
   end
   <-------> object Copy1: TMenuItem
    Caption = '&Copy...'
    ShortCut = 119
    OnClick = FileChange
   end
   <-------> object Delete1: TMenuItem
    Caption = '&Delete...'
    ShortCut = 46
    OnClick = Delete1Click
   end
   <-------> object Rename1: TMenuItem
    Caption = '&Rename...'
    OnClick = FileChange
   end
   <-------> object Properties1: TMenuItem
    Caption = '&Properties...'
    ShortCut = 32781
    OnClick = Properties1Click
   end
   <-------> object N1: TMenuItem
    Caption = '-'
   end
```

```
  Height = 21
  Align = alBottom
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -12
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  Style = tsOwnerDraw
  OnChange = DriveTabSetChange
  OnDrawTab = DriveTabSetDrawTab
  OnMeasureTab = DriveTabSetMeasureTab
 end
 <-------> object DirectoryOutline: TDirectoryOutline
  Left = 0
  Top = 0
  Width = 161
  Height = 631
  Align = alLeft
  ItemHeight = 13
  Options = [ooDrawFocusRect]
  PictureLeaf.Data = {
  TabOrder = 2
  OnChange = DirectoryOutlineChange
  OnDragDrop = DirectoryOutlineDragDrop
  OnDragOver = DirectoryOutlineDragOver
  Data = {10}
 end
 <-------> object FileList: TFileListBox
  Left = 161
  Top = 0
  Width = 863
  Height = 631
  Align = alClient
  FileType = [ftReadOnly, ftDirectory, ftArchive, ftNormal]
  ItemHeight = 16
  ShowGlyphs = True
  TabOrder = 3
  OnChange = FileListChange
  OnEndDrag = FileListEndDrag
```

```
  end
  <-------> object ChangeDate:
TLabel
    Left = 24
    Top = 64
    Width = 60
    Height = 13
    Caption = 'ChangeDate'
  end
  <-------> object OKBtn: TButton
    Left = 300
    Top = 8
    Width = 77
    Height = 27
    Caption = 'OK'
    Default = True
    ModalResult = 1
    TabOrder = 0
  end
  <-------> object CancelBtn: TButton
    Left = 300
    Top = 40
    Width = 77
    Height = 27
    Cancel = True
    Caption = 'Cancel'
    ModalResult = 2
    TabOrder = 1
  end
  <-------> object GroupBox1:
TGroupBox
    Left = 24
    Top = 88
    Width = 249
    Height = 65
    Caption = 'Attributes'
    TabOrder = 2
    <-------> object ReadOnly:
TCheckBox
      Left = 16
      Top = 16
      Width = 97
      Height = 17
      Caption = '&Read Only'
      TabOrder = 0
    end
    <-------> object Archive:
TCheckBox
```

```
      <-------> object Exit1:
TMenuItem
        Caption = 'E&xit'
        OnClick = Exit1Click
      end
    end
  end
End
\\\\\\\\\\\\\\\\\\\\\\

<-------> object FileAttrForm:
TFileAttrForm
  Left = 180
  Top = 118
  ActiveControl = OKBtn
  BorderStyle = bsDialog
  Caption = 'File attributes'
  ClientHeight = 179
  ClientWidth = 388
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Bevel1: TBevel
    Left = 8
    Top = 8
    Width = 281
    Height = 161
    Shape = bsFrame
    IsControl = True
  end
  <-------> object FileDirName:
TLabel
    Left = 24
    Top = 16
    Width = 57
    Height = 13
    Caption = 'FileDirName'
  end
  <-------> object FilePathName:
TLabel
    Left = 24
    Top = 40
    Width = 66
    Height = 13
    Caption = 'FilePathName'
```

```
  Top = 8
  Width = 281
  Height = 105
  Shape = bsFrame
  IsControl = True
end
<-------> object CurrentDir: TLabel
  Left = 24
  Top = 24
  Width = 47
  Height = 13
  Caption = 'CurrentDir'
end
<-------> object Label1: TLabel
  Left = 24
  Top = 48
  Width = 26
  Height = 13
  Caption = 'From:'
end
<-------> object Label2: TLabel
  Left = 24
  Top = 80
  Width = 16
  Height = 13
  Caption = 'To:'
end
<-------> object OKBtn: TButton
  Left = 300
  Top = 8
  Width = 77
  Height = 27
  Caption = 'OK'
  Default = True
  ModalResult = 1
  TabOrder = 0
end
<-------> object CancelBtn: TButton
  Left = 300
  Top = 40
  Width = 77
  Height = 27
  Cancel = True
  Caption = 'Cancel'
  ModalResult = 2
  TabOrder = 1
end
<-------> object ToFileName: TEdit
```

```
    Left = 16
    Top = 40
    Width = 97
    Height = 17
    Caption = '&Archive'
    TabOrder = 1
  end
  <-------> object System:
TCheckBox
    Left = 120
    Top = 16
    Width = 97
    Height = 17
    Caption = '&System'
    TabOrder = 2
  end
  <-------> object Hidden:
TCheckBox
    Left = 120
    Top = 40
    Width = 97
    Height = 17
    Caption = '&Hidden'
    TabOrder = 3
  end
 end
End

\=\
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

<-------> object ChangeDlg:
TChangeDlg
 Left = 188
 Top = 338
 ActiveControl = OKBtn
 BorderStyle = bsDialog
 Caption = 'Dialog'
 ClientHeight = 123
 ClientWidth = 388
 Color = clBtnFace
 ParentFont = True
 OldCreateOrder = True
 Position = poScreenCenter
 PixelsPerInch = 96
 TextHeight = 13
 <-------> object Bevel1: TBevel
  Left = 8
```

```
    Left = 64
    Top = 80
    Width = 209
    Height = 21
    TabOrder = 2
  end
  <-------> object FromFileName:
TEdit
    Left = 64
    Top = 48
    Width = 209
    Height = 21
    TabOrder = 3
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

الملف التنفيذي لبرنامج

42 procedures

ResXplor)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit RXMain;

{ This program provides an example of how to use the TreeView and ListView
  components in a fashion similar to the Microsoft Windows Explorer.

  It is not intended to be a fully functional resource viewer. }

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls, Forms, Dialogs,
  ExeImage, StdCtrls, Buttons, ExtCtrls, ComCtrls, Menus,
RXMisc, HexDump,
  ImgList;

type
  TMainForm = class(TForm)
    StatusBar: TStatusBar;
    TreeViewPanel: TPanel;
    Panel1: TPanel;
    ImageViewer: TImage;
    ListView: TListView;
    TreeView: TTreeView;
    Splitter: TPanel;
    Notebook: TNotebook;
    ListViewPanel: TPanel;
    ListViewCaption: TPanel;
    FileOpenDialog: TOpenDialog;
    FileSaveDialog: TSaveDialog;
    MainMenu: TMainMenu;
    miFile: TMenuItem;
    miFileExit: TMenuItem;
    miFileSave: TMenuItem;
    miFileOpen: TMenuItem;
    miView: TMenuItem;
    miViewStatusBar: TMenuItem;
    miViewLargeIcons: TMenuItem;
    miViewSmallIcons: TMenuItem;
    miViewList: TMenuItem;
    miViewDetails: TMenuItem;
    miHelp: TMenuItem;
    miHelpAbout: TMenuItem;
    Small: TImageList;
    Large: TImageList;
    StringViewer: TMemo;
    <-------> procedure FileExit(Sender: TObject);
    <-------> procedure FileOpen(Sender: TObject);
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure ListViewEnter(Sender: TObject);
    <-------> procedure SaveResource(Sender: TObject);
    <-------> procedure SelectListViewType(Sender: TObject);
    <-------> procedure ShowAboutBox(Sender: TObject);
```

```

uses About, RXTypes;

{$R *.dfm}
{$R RXIMAGES.RES}

const
  itBitmap: TResType = ImgList.rtBitmap; // Reference for duplicate identifier
  ImageMap: array[TResourceType] of Byte = (2,4,5,3,2,2,2,2,2,2,2,2,2,2,2,2,2);
  ResFiltMap: array[TResourceType] of Byte = (1,4,2,3,1,1,1,1,1,1,1,1,1,1,1,1,1);

  SNoResSelected = 'No resource selected';
  SFormCaption   = 'Resource Explorer';
  SSaveFilter    = 'Other Resource (*.*)|*.*|Bitmaps (*.BMP)|*.BMP|'+
                   'Icons (*.ICO)|*.ICO|Cursor (*.CUR)|*.CUR';
  SOpenFilter    = 'Executable File Images (*.EXE;*.DLL)|*.EXE;*.DLL|'+
                   'All Files (*.*)|*.*';

{ Utility Functions }

<-------> procedure Error(const ErrMsg: string);
begin
  raise Exception.Create(ErrMsg);
end;

<-------> procedure ErrorFmt(const ErrMsg: string; Params: array of const);
begin
  raise Exception.Create(format(ErrMsg, Params));
end;

function Confirm(const AMsg:
```

```
    <-------> procedure ToggleStatusBar(Sender: TObject);
    <-------> procedure TreeViewChange(Sender: TObject; Node: TTreeNode);
    <-------> procedure SplitterMouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    <-------> procedure SplitterMouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    <-------> procedure SplitterMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    <-------> procedure ViewMenuDropDown(Sender: TObject);
    <-------> procedure NotebookEnter(Sender: TObject);
    <-------> procedure FormDestroy(Sender: TObject);
  private
    FExeFile: TExeImage;
    HexDump: THexDump;
    SplitControl: TSplitControl;
    <-------> procedure LoadResources(ResList: TResourceList; Node: TTreeNode);
    <-------> procedure DisplayResources;
    <-------> procedure UpdateViewPanel;
    <-------> procedure UpdateListView(ResList: TResourceList);
    <-------> procedure UpdateStatusLine(ResItem: TResourceItem);
  end;

var
  MainForm: TMainForm;

implementation
```

```
FileName)]);
  with TreeView do
  begin
    SetFocus;
    Selected := Items[0];
  end;
end;

<-------> procedure TMainForm.UpdateViewPanel;
var
  R: TResourceItem;
begin
  with TreeView do
  begin
    if Visible and Assigned(Selected) then
    begin
      R := TResourceItem(Selected.Data);
      if R.IsList then UpdateListView(R.List) else
      begin
        case R.ResType of
          rtBitmap, rtIconEntry, rtCursorEntry:
            begin
              ImageViewer.Picture.Assign(R);
              Notebook.PageIndex := 1;
            end;
          rtString, rtMenu:
            begin
              StringViewer.Lines.Assign(R);
              StringViewer.SelStart := 0;
              Notebook.PageIndex := 2;
            end
          else
            begin
              HexDump.Address := R.RawData;
              HexDump.DataSize := R.Size;
              Notebook.PageIndex := 3;
            end;
        end;
      end;
```

```
String): Boolean;
begin
  Result := MessageDlg(AMsg, mtConfirmation, mbYesNoCancel, 0) = idYes;
end;

{ Non Event Handlers }

<-------> procedure TMainForm.LoadResources(ResList : TResourceList; Node: TTreeNode);
var
  I: Integer;
  CNode: TTreeNode;
begin
  for I := 0 to ResList.Count - 1 do
    with ResList[I] do
    begin
      CNode := TreeView.Items.AddChildObject(Node, Name, ResList[I]);
      if IsList then
      begin
        CNode.SelectedIndex := 1;
        LoadResources(List, CNode);
      end else
      begin
        CNode.ImageIndex := ImageMap[ResList[I].ResType];
        CNode.SelectedIndex := CNode.ImageIndex;
      end;

    end;
end;

<-------> procedure TMainForm.DisplayResources;
begin
  ListView.Items.Clear;
  TreeView.Selected := nil;
  TreeView.Items.Clear;
  LoadResources(FExeFile.Resources, nil);
  Caption := Format('%s - %s', [SFormCaption, AnsiLowerCaseFileName(FExeFile.
```

```
    StatusBar.Panels[1].Text :=
Format(' Offset: %x  Size: %x',
[Offset, Size]);
  end;
end;

{ Form Initialization }

<-------> procedure
TMainForm.FormCreate(Sender:
TObject);
begin
  SplitControl :=
TSplitControl.Create(Self);
  HexDump :=
CreateHexDump(TWinControl(Note
Book.Pages.Objects[3]));
  FileOpenDialog.Filter :=
SOpenFilter;
  FileSaveDialog.Filter :=
SSaveFilter;
  Small.ResourceLoad(itBitmap,
'SmallImages', clOlive);
  Large.ResourceLoad(itBitmap,
'LargeImages', clOlive);
  Notebook.PageIndex := 0;
  if (ParamCount > 0) and
FileExists(ParamStr(1)) then
  begin
    Show;
    FExeFile :=
TExeImage.CreateImage(Self,
ParamStr(1));
    DisplayResources;
  end;
end;

{ Menu Event Handlers }

<-------> procedure
TMainForm.FileOpen(Sender:
TObject);
var
  TmpExeFile: TExeImage;
begin
  with FileOpenDialog do
  begin
    if not Execute then Exit;
    TmpExeFile :=
```

```
      UpdateStatusLine(R);
    end;
  end;
end;

<-------> procedure
TMainForm.UpdateListView(ResLis
t: TResourceList);
var
  I: Integer;
begin
  ListView.Items.Clear;
  for I := 0 to ResList.Count-1 do
    with ResList[I],
ListView.Items.Add do
    begin
      Data := ResList[I];
      Caption := Name;
      SubItems.Add(Format('%.7x',
[Offset]));
      SubItems.Add(Format('%.5x',
[Size]));
      ImageIndex :=
ImageMap[ResType];
    end;
  Notebook.PageIndex := 0;
end;

<-------> procedure
TMainForm.UpdateStatusLine(ResI
tem: TResourceItem);
begin
  if ResItem.IsList then
  begin
    ListViewCaption.Caption := '
'+TreeView.Selected.Text;
    StatusBar.Panels[0].Text :=
Format(' %d object(s)',
[ListView.Items.Count]);
    StatusBar.Panels[1].Text :=
Format(' Offset: %x',
[ResItem.Offset]);
  end else
  with ResItem do
  begin
    ListViewCaption.Caption :=
Format(' %s: %s', [ResTypeStr,
Name]);
    StatusBar.Panels[0].Text := '';
```

```
ata);
  end;

  function
ListViewResourceSelected: Boolean;
  begin
    Result :=
Assigned(ListView.Selected) and

Assigned(ListView.Selected.Data)
and
          not
TResourceItem(ListView.Selected.D
ata).IsList;
    if Result then ResItem :=
TResourceItem(ListView.Selected.D
ata);
  end;

begin
  if TreeViewResourceSelected or
ListViewResourceSelected then
  with FileSaveDialog do
  begin
    FilterIndex :=
ResFiltMap[ResItem.ResType];
    if Execute then
      ResItem.SaveToFile(FileName)
  end
  else
    Error(SNoResSelected);
end;

<-------> procedure
TMainForm.SelectListViewType(Se
nder: TObject);
begin
  ListView.ViewStyle :=
TViewStyle(TComponent(Sender).T
ag);
end;

<-------> procedure
TMainForm.ShowAboutBox(Sender
: TObject);
begin
  About.ShowAboutBox;
end;

```

```
TExeImage.CreateImage(Self,
FileName);
    if Assigned(FExeFile) then
FExeFile.Destroy;
    FExeFile := TmpExeFile;
    DisplayResources;
  end;
end;

<-------> procedure
TMainForm.FileExit(Sender:
TObject);
begin
  Close;
end;

<-------> procedure
TMainForm.ListViewEnter(Sender:
TObject);
begin
  with ListView do
    if (Items.Count > 1) and (Selected
= nil) then
    begin
      Selected := Items[0];
      ItemFocused := Selected;
    end;
end;

<-------> procedure
TMainForm.SaveResource(Sender:
TObject);
var
  ResItem: TResourceitem;

  function
TreeViewResourceSelected:
Boolean;
  begin
    Result :=
Assigned(TreeView.Selected) and

Assigned(TreeView.Selected.Data)
and
          not
TResourceItem(TreeView.Selected.D
ata).IsList;
    if Result then ResItem :=
TResourceItem(TreeView.Selected.D
```

```
end;

<-------> procedure TMainForm.SplitterMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  with SplitControl do if Sizing then EndSizing;
end;

<-------> procedure TMainForm.NotebookEnter(Sender: TObject);
var
  Page: TWinControl;
begin
  with NoteBook do
  begin
    Page := TWinControl(Pages.Objects[PageIndex]);
    if (Page.ControlCount > 0) and (Page.Controls[0] is TWinControl) then
      TWinControl(Page.Controls[0]).SetFocus;
  end;
end;

<-------> procedure TMainForm.FormDestroy(Sender: TObject);
begin
  SplitControl.Free;
end;

end.
```

<-------> object35
الملف النصي لبرنامج
ResXplor)
إعداد علاء الدين اللباد

```
<-------> procedure TMainForm.ToggleStatusBar(Sender: TObject);
begin
  StatusBar.Visible := not StatusBar.Visible;
end;

<-------> procedure TMainForm.TreeViewChange(Sender: TObject; Node: TTreeNode);
begin
  UpdateViewPanel;
end;

<-------> procedure TMainForm.ViewMenuDropDown(Sender: TObject);
var
  I: Integer;
begin
  miViewStatusBar.Checked := StatusBar.Visible;
  for I := 0 to miView.Count-1 do
    with miView.Items[I] do
      if GroupIndex = 1 then
        Checked := (Tag = Ord(ListView.ViewStyle));
end;

<-------> procedure TMainForm.SplitterMouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  if (Button = mbLeft) and (Shift = [ssLeft]) then
    SplitControl.BeginSizing(Splitter, TreeViewPanel);
end;

<-------> procedure TMainForm.SplitterMouseMove(Sender: TObject; Shift: TShiftState;
  X, Y: Integer);
begin
  with SplitControl do if Sizing then ChangeSizing(X, Y);
```

هاتف ٠٩٤٤٥٧٥٣٧١

```
<-------> object MainForm: TMainForm
  Left = -4
  Top = -4
  Width = 1032
  Height = 746
  Caption = 'Resource Explorer'
  Color = clBtnFace
  ParentFont = True
  Icon.Data = {
    0FF}
  Menu = MainMenu
  OldCreateOrder = True
  OnCreate = FormCreate
  OnDestroy = FormDestroy
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object StatusBar: TStatusBar
    Left = 0
    Top = 673
    Width = 1024
    Height = 19
    Panels = <
      item
        Width = 150
      end
      item
        Width = 500
      end>
  end
  <-------> object TreeViewPanel: TPanel
    Left = 0
    Top = 0
    Width = 185
    Height = 673
    Align = alLeft
    BevelOuter = bvNone
    TabOrder = 1
    <-------> object Panel1: TPanel
      Left = 0
      Top = 0
      Width = 185
      Height = 21
      Align = alTop
      Alignment = taLeftJustify
      BevelInner = bvLowered
      BevelOuter = bvNone
      BorderWidth = 1
      Caption = ' Resources'
      TabOrder = 0
    end
    <-------> object TreeView: TTreeView
      Left = 0
      Top = 21
      Width = 185
      Height = 652
      Align = alClient
      Images = Small
      Indent = 19
      ReadOnly = True
      TabOrder = 1
      OnChange = TreeViewChange
    end
  end
  <-------> object Splitter: TPanel
    Left = 185
    Top = 0
    Width = 2
    Height = 673
    Cursor = crHSplit
    Align = alLeft
    BevelOuter = bvNone
    TabOrder = 2
    OnMouseDown = SplitterMouseDown
    OnMouseMove = SplitterMouseMove
    OnMouseUp = SplitterMouseUp
  end
  <-------> object ListViewPanel: TPanel
    Left = 187
    Top = 0
    Width = 837
    Height = 673
    Align = alClient
    BevelOuter = bvNone
    TabOrder = 3
    <-------> object ListViewCaption: TPanel
      Left = 0
```

```
      ReadOnly = True
      SmallImages = Small
      TabOrder = 0
      ViewStyle = vsReport
      OnEnter = ListViewEnter
    end
  end
  <-------> object TPage
    Left = 0
    Top = 0
    Caption = 'ImageViewPage'
    <-------> object ImageViewer: TImage
      Left = 0
      Top = 0
      Width = 837
      Height = 652
      Align = alClient
      Center = True
    end
  end
  <-------> object TPage
    Left = 0
    Top = 0
    Caption = 'StringViewPage'
    <-------> object StringViewer: TMemo
      Left = 0
      Top = 0
      Width = 382
      Height = 175
      Align = alClient
      ReadOnly = True
      ScrollBars = ssBoth
      TabOrder = 0
      WantReturns = False
      WordWrap = False
    end
  end
  <-------> object TPage
    Left = 0
    Top = 0
    Caption = 'OtherViewPage'
  end
 end
end
<-------> object MainMenu: TMainMenu
```

```
    Top = 0
    Width = 837
    Height = 21
    Align = alTop
    Alignment = taLeftJustify
    BevelInner = bvLowered
    BevelOuter = bvNone
    BorderWidth = 1
    TabOrder = 0
  end
  <-------> object Notebook: TNotebook
    Left = 0
    Top = 21
    Width = 837
    Height = 652
    Align = alClient
    Color = clBtnFace
    PageIndex = 1
    ParentColor = False
    TabOrder = 1
    OnEnter = NotebookEnter
    <-------> object TPage
      Left = 0
      Top = 0
      Caption = 'ListViewPage'
      <-------> object ListView: TListView
        Left = 0
        Top = 0
        Width = 382
        Height = 175
        Align = alClient
        Columns = <
          item
            Caption = 'Name'
            Width = 150
          end
          item
            Caption = 'Offset'
            Width = 80
          end
          item
            Caption = 'Size'
            Width = 80
          end>
        ColumnClick = False
        LargeImages = Large
```

```
      GroupIndex = 1
      RadioItem = True
      OnClick = SelectListViewType
    end
    <-------> object miViewList: TMenuItem
      Tag = 2
      Caption = '&List'
      GroupIndex = 1
      RadioItem = True
      OnClick = SelectListViewType
    end
    <-------> object miViewDetails: TMenuItem
      Tag = 3
      Caption = '&Details'
      GroupIndex = 1
      RadioItem = True
      OnClick = SelectListViewType
    end
  end
  <-------> object miHelp: TMenuItem
    Caption = '&Help'
    <-------> object miHelpAbout: TMenuItem
      Caption = '&About'
      OnClick = ShowAboutBox
    end
  end
 end
 <-------> object FileOpenDialog: TOpenDialog
   Options = [ofOverwritePrompt, ofHideReadOnly, ofPathMustExist]
   Left = 497
   Top = 161
 end
 <-------> object FileSaveDialog: TSaveDialog
   Options = [ofOverwritePrompt, ofHideReadOnly, ofPathMustExist]
   Left = 463
   Top = 161
 end
 <-------> object Small: TImageList
   Left = 384
   Top = 161
```

```
   Left = 530
   Top = 161
   <-------> object miFile: TMenuItem
     Caption = '&File'
     <-------> object miFileOpen: TMenuItem
       Caption = '&Open...'
       OnClick = FileOpen
     end
     <-------> object miFileSave: TMenuItem
       Caption = '&Save Resource...'
       OnClick = SaveResource
     end
     <-------> object miN1: TMenuItem
       Caption = '-'
     end
     <-------> object miFileExit: TMenuItem
       Caption = 'E&xit'
       OnClick = FileExit
     end
   end
   <-------> object miView: TMenuItem
     Caption = '&View'
     OnClick = ViewMenuDropDown
     <-------> object miViewStatusBar: TMenuItem
       Caption = 'Status &Bar'
       OnClick = ToggleStatusBar
     end
     <-------> object N6: TMenuItem
       Caption = '-'
     end
     <-------> object miViewLargeIcons: TMenuItem
       Caption = 'Lar&ge Icons'
       GroupIndex = 1
       RadioItem = True
       OnClick = SelectListViewType
     end
     <-------> object miViewSmallIcons: TMenuItem
       Tag = 1
       Caption = 'S&mall Icons'
```

```
  DBCtrls, StdCtrls, ExtCtrls, Mask,
DBTables, DB, Grids, DBGrids,
Menus,
  DBCGrids;

type
  TFmCtrlGrid = class(TForm)
    DBCtrlGrid1: TDBCtrlGrid;
    DBGrid1: TDBGrid;
    DBEdit1: TDBEdit;
    DBEdit2: TDBEdit;
    DBEdit3: TDBEdit;
    DBEdit4: TDBEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Button1: TButton;
    Bevel1: TBevel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    lPurchase: TLabel;
    lTotalCost: TLabel;
    lTotalShares: TLabel;
    lDifference: TLabel;
    DBNavigator1: TDBNavigator;
    MainMenu1: TMainMenu;
    About1: TMenuItem;
    <----procedure
Button1Click(Sender: TObject);
    <----procedure
FormShow(Sender: TObject);
    <----procedure
DBGrid1Enter(Sender: TObject);
    <----procedure
DBCtrlGrid1Enter(Sender:
TObject);
    <----procedure
About1Click(Sender: TObject);
    <----procedure
FormCreate(Sender: TObject);
  private
    { Private declarations }
```

```
  end
  <-------> object Large: TImageList
    Height = 32
    Width = 32
    Left = 417
    Top = 161
  end
end
```

القسم الخامس
برمجة قواعد البيانات
في لغة البرمجة دلفي

البرنامج يتعامل مع أكثر الإجراءات المتعلقة بقواعد البيانات وهو موجود في دلفي٧ واسمه :

CtrlGrid

من إعداد

علاء الدين اللباد
٠٩٤٤٥٧٥٣٧١

يحتوي البرنامج على24Procedure
يتعامل مع قواعد البيانات
الملف التنفيذي للبرنامج للوحدةCtrlGrid

```
unit CtrlForm;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
```

```
    ShowModal;
  finally
    Free;
  end;
end;

<----procedure
TFmCtrlGrid.FormCreate(Sender:
TObject);
begin

end;


end.


```

الملف التنفيذي للوحدةabout

```
unit About;

interface

uses Windows, SysUtils, Classes,
Graphics, Forms, Controls, StdCtrls,
  Buttons, ExtCtrls;

type
  TFMAboutBox = class(TForm)
    Panel1: TPanel;
    ProgramIcon: TImage;
    ProductName: TLabel;
    Version: TLabel;
    OKButton: TButton;
    Memo1: TMemo;
    <----procedure
FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FMAboutBox: TFMAboutBox;

implementation
```

```
  public
    { Public declarations }
  end;

var
  FmCtrlGrid: TFmCtrlGrid;

implementation

uses DM, About;

{$R *.dfm}

<----procedure
TFmCtrlGrid.Button1Click(Sender:
TObject);
begin
  Close;
end;

<----procedure
TFmCtrlGrid.FormShow(Sender:
TObject);
begin
  DM1.CalculateTotals(Sender, nil);
end;

<----procedure
TFmCtrlGrid.DBGrid1Enter(Sende
r: TObject);
begin
  DBNavigator1.DataSource :=
DM1.DSMaster;
end;

<----procedure
TFmCtrlGrid.DBCtrlGrid1Enter(Se
nder: TObject);
begin
  DBNavigator1.DataSource :=
DM1.DSHoldings;
end;

<----procedure
TFmCtrlGrid.About1Click(Sender:
TObject);
begin
  with TFMAboutBox.Create(nil) do
  try
```

```
    tblMasterINDUSTRY:
TSmallintField;
    tblMasterPRICE_CHG:
TSmallintField;
    tblMasterRATING: TStringField;
    tblMasterRANK: TFloatField;
    tblMasterOUTLOOK:
TSmallintField;
    tblMasterRCMNDATION:
TStringField;
    tblMasterRISK: TStringField;
    dsMaster: TDataSource;
    tblIndustry: TTable;
    tblIndustryIND_CODE:
TSmallintField;
    tblIndustryIND_NAME:
TStringField;
    tblIndustryLONG_NAME:
TStringField;
    dsIndustry: TDataSource;
    tblHoldings: TTable;
    tblHoldingsACCT_NBR:
TFloatField;
    tblHoldingsSHARES:
TFloatField;
    tblHoldingsPUR_PRICE:
TFloatField;
    tblHoldingsPUR_DATE:
TDateField;
    tblHoldingsSYMBOL:
TStringField;
    tblHoldingsPUR_COST:
TCurrencyField;
    dsHoldings: TDataSource;
    <----procedure
tblHoldingsCalcFields(DataSet:
TDataSet);
    <----procedure
tblHoldingsAfterPost(DataSet:
TDataSet);
    <----procedure
CalculateTotals(Sender: TObject;
Field: TField);
    <----procedure
tblHoldingsAfterOpen(DataSet:
TDataSet);
    <----procedure
DataModuleCreate(Sender:
TObject);
```

```

{$R *.dfm}

<----procedure
TFMAboutBox.FormCreate(Sender:
TObject);
begin

end;

end.



```

الملف التنفيذي للوحدة dm

```


unit DM;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  DBTables, DB;

type
  TDM1 = class(TDataModule)
    tblMaster: TTable;
    tblMasterSYMBOL:
TStringField;
    tblMasterCO_NAME:
TStringField;
    tblMasterEXCHANGE:
TStringField;
    tblMasterIndustryLongName:
TStringField;
    tblMasterCUR_PRICE:
TFloatField;
    tblMasterYRL_HIGH:
TFloatField;
    tblMasterYRL_LOW:
TFloatField;
    tblMasterP_E_RATIO:
TFloatField;
    tblMasterPROJ_GRTH:
TFloatField;
```

```
displays; otherwise,
  clear the result displays. }
 if tblHoldings.recordCount = 0 then
 begin
  { Clear the result displays }
  FmCtrlGrid.lTotalCost.Caption
:= '';
  FmCtrlGrid.lTotalShares.Caption
:= '';
  FmCtrlGrid.lDifference.Caption
:= '';
 end
 else
 begin
  { let the user know something's
going on }
  Screen.Cursor := crHourglass;

  { Initialize the holder variables }
  flTotalCost := 0.0;
  flTotalShares := 0.0;

  { Calculate the total cost of these
holdings. }
  tblHoldings.disableControls;  {
hide this process from the user }
  tblHoldings.first;
  while not tblHoldings.eof do
  begin
    flTotalCost := flTotalCost +
tblHoldingsPUR_COST.AsFloat;
    flTotalShares := flTotalShares +
tblHoldingsSHARES.AsFloat;
    tblHoldings.next;
  end;
  tblHoldings.first;
  tblHoldings.enableControls;  {
restore the display of holdings }

  { Calculate the current value of
the shares (by multiplying
    the current holdings by the
current share price) and the
    difference between the cost and
the value. }

  flTotalValue := flTotalShares *
tblMasterCUR_PRICE.AsFloat;
  flDifference := flTotalValue -
```

```
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  DM1: TDM1;

implementation

uses CtrlForm;

{$R *.dfm}

<----procedure
TDM1.tblHoldingsCalcFields(DataS
et: TDataSet);
begin
  tblHoldingsPUR_COST.AsFloat :=
    tblHoldingsPUR_PRICE.AsFloat
* tblHoldingsSHARES.AsFloat;
end;

<----procedure
TDM1.CalculateTotals(Sender:
TObject; Field: TField);
var
  flTotalCost,          { Holds total
share cost }
  flTotalShares,         { Holds total
share count }
  flTotalValue,          { Holds total
share value }
  flDifference: Real;     { Holds
difference between cost and value }
  strFormatSpec: string;  { The
Display Format specification }
begin

  { Update the count of stock
transactions }
  FmCtrlGrid.lPurchase.Caption :=
IntToStr( tblHoldings.RecordCount
);

  { See whether or not its necessary
to total the holdings and
    (if so) do so and update the result
```

```
flTotalCost;

   { Use the same format
specification as that being used to
     display the Current Price field
value so it can be used
     to display the results }

   strFormatSpec :=
tblMasterCUR_PRICE.DisplayFor
mat;

   { Update the result displays }

   FmCtrlGrid.lTotalCost.Caption
:=
      FormatFloat( strFormatSpec,
flTotalCost );
   FmCtrlGrid.lTotalShares.Caption
:=
      FormatFloat( strFormatSpec,
flTotalValue );
   FmCtrlGrid.lDifference.Caption
:=
      FormatFloat( strFormatSpec,
flDifference );

   { Update the Font Color of the
Diference to
     indicate the quality of the
investment }
   if flDifference > 0 then

FmCtrlGrid.lDifference.Font.Color
:= clGreen
   else

FmCtrlGrid.lDifference.Font.Color
:= clRed;
   FmCtrlGrid.lDifference.update;

   { let the user know that we're
finished }
   Screen.Cursor := crDefault;
  end;
end;


<----procedure
TDM1.tblHoldingsAfterPost(DataSe
t: TDataSet);
var
  bmCurrent : TBookmark;  { Holds
the current position }
begin
  with tblHoldings do
  begin
    bmCurrent := getBookmark;       {
save position }
    try
      CalculateTotals(nil, nil);  { recalc
totals }
      gotoBookmark(bmCurrent);     {
restore position }
    finally;
      freeBookmark(bmCurrent);     {
free memory }
    end;
  end;
end;

<----procedure
TDM1.tblHoldingsAfterOpen(DataS
et: TDataSet);
begin
  {Don't want this calculation to
occur until both master & detail are
open}
  dsMaster.OnDataChange :=
CalculateTotals;
end;

<----procedure
TDM1.DataModuleCreate(Sender:
TObject);
begin

end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

الملف النصي لبرنامج

Object٦٦

(CtrlGri)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

CtrlGridالملف النصي لواجهة البرنامج

```
<------->object FmCtrlGrid: TFmCtrlGrid
  Left = 221
  Top = 170
  ActiveControl = DBGrid1
  BorderStyle = bsSingle
  Caption = 'Stock Browser'
  ClientHeight = 340
  ClientWidth = 586
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = [fsBold]
  Menu = MainMenu1
  OldCreateOrder = True
  OnCreate = FormCreate
  OnShow = FormShow
  PixelsPerInch = 96
  TextHeight = 13
  <------->object Bevel1: TBevel
    Left = 4
    Top = 212
    Width = 220
    Height = 85
    Shape = bsFrame
  end
  <------->object Label5: TLabel
    Left = 6
    Top = 4
    Width = 61
    Height = 13
    AutoSize = False
    Caption = '&Securities:'
    FocusControl = DBGrid1
  end
  <------->object Label6: TLabel
    Left = 230
    Top = 152
    Width = 54
    Height = 13
    AutoSize = False
    Caption = '&Holdings:'
    FocusControl = DBEdit2
  end
  <------->object Label7: TLabel
    Left = 4
    Top = 196
    Width = 103
    Height = 13
    AutoSize = False
    Caption = 'Investment Value:'
  end
  <------->object Label8: TLabel
    Left = 12
    Top = 220
    Width = 105
    Height = 13
    Alignment = taRightJustify
    Caption = 'Number of Purchases:'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -12
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
  end
  <------->object Label9: TLabel
    Left = 15
    Top = 242
    Width = 82
    Height = 13
    Alignment = taRightJustify
    Caption = 'Total Share Cost:'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -12
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
  end
  <------->object Label10: TLabel
```

```
   Width = 110
   Height = 13
   Alignment = taRightJustify
   AutoSize = False
  end
  <------->object lDifference: TLabel
   Left = 103
   Top = 279
   Width = 110
   Height = 13
   Alignment = taRightJustify
   AutoSize = False
  end
  <------->object DBCtrlGrid1:
TDBCtrlGrid
   Left = 228
   Top = 168
   Width = 351
   Height = 159
   DataSource = DM1.dsHoldings
   PanelHeight = 53
   PanelWidth = 334
   TabOrder = 1
   OnEnter = DBCtrlGrid1Enter
   <------->object Label1: TLabel
    Left = 8
    Top = 9
    Width = 74
    Height = 13
    Alignment = taRightJustify
    Caption = 'Purchase Date:'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -12
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
   end
   <------->object Label2: TLabel
    Left = 183
    Top = 32
    Width = 75
    Height = 13
    Alignment = taRightJustify
    Caption = 'Purchase Price:'
    Font.Charset =
DEFAULT_CHARSET
   Left = 19
   Top = 260
   Width = 78
   Height = 13
   Alignment = taRightJustify
   Caption = 'Value of Shares:'
   Font.Charset =
DEFAULT_CHARSET
   Font.Color = clWindowText
   Font.Height = -12
   Font.Name = 'MS Sans Serif'
   Font.Style = []
   ParentFont = False
  end
  <------->object Label11: TLabel
   Left = 23
   Top = 279
   Width = 74
   Height = 13
   Alignment = taRightJustify
   Caption = 'C/V Difference:'
   Font.Charset =
DEFAULT_CHARSET
   Font.Color = clWindowText
   Font.Height = -12
   Font.Name = 'MS Sans Serif'
   Font.Style = []
   ParentFont = False
  end
  <------->object lPurchase: TLabel
   Left = 123
   Top = 220
   Width = 86
   Height = 13
   AutoSize = False
  end
  <------->object lTotalCost: TLabel
   Left = 103
   Top = 242
   Width = 110
   Height = 13
   Alignment = taRightJustify
   AutoSize = False
  end
  <------->object lTotalShares:
TLabel
   Left = 103
   Top = 260
```

```
    Width = 90
    Height = 21
    DataField = 'PUR_DATE'
    DataSource = DM1.dsHoldings
    TabOrder = 1
  end
  <------->object DBEdit3: TDBEdit
    Left = 259
    Top = 29
    Width = 66
    Height = 21
    DataField = 'PUR_PRICE'
    DataSource = DM1.dsHoldings
    TabOrder = 2
  end
  <------->object DBEdit4: TDBEdit
    Left = 259
    Top = 6
    Width = 66
    Height = 21
    DataField = 'SHARES'
    DataSource = DM1.dsHoldings
    TabOrder = 3
  end
 end
 <------->object DBGrid1: TDBGrid
  Left = 4
  Top = 20
  Width = 573
  Height = 128
  DataSource = DM1.dsMaster
  Options = [dgTitles, dgIndicator,
dgColumnResize, dgColLines,
dgRowLines, dgTabs, dgRowSelect,
dgAlwaysShowSelection,
dgConfirmDelete, dgCancelOnExit]
  TabOrder = 0
  TitleFont.Charset =
DEFAULT_CHARSET
  TitleFont.Color = clWindowText
  TitleFont.Height = -11
  TitleFont.Name = 'MS Sans Serif'
  TitleFont.Style = [fsBold]
  OnEnter = DBGrid1Enter
  Columns = <
   item
    Expanded = False
    FieldName = 'SYMBOL'
```

```
    Font.Color = clWindowText
    Font.Height = -12
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
  end
  <------->object Label3: TLabel
    Left = 185
    Top = 9
    Width = 73
    Height = 13
    Alignment = taRightJustify
    Caption = 'Shares Bought:'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -12
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
  end
  <------->object Label4: TLabel
    Left = 17
    Top = 32
    Width = 65
    Height = 13
    Alignment = taRightJustify
    Caption = 'Acct Number:'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -12
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
  end
  <------->object DBEdit1: TDBEdit
    Left = 83
    Top = 29
    Width = 90
    Height = 21
    DataField = 'ACCT_NBR'
    DataSource = DM1.dsHoldings
    TabOrder = 0
  end
  <------->object DBEdit2: TDBEdit
    Left = 83
    Top = 6
```

```
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'PRICE_CHG'
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'RATING'
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'RANK'
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'OUTLOOK'
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'RCMNDATION'
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'RISK'
    Visible = True
  end>
end
<------->object Button1: TButton
  Left = 82
  Top = 309
  Width = 75
  Height = 24
  Caption = '&Close'
  TabOrder = 2
  OnClick = Button1Click
end
```

```
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'CO_NAME'
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName =
'IndustryLongName'
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'EXCHANGE'
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'CUR_PRICE'
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'YRL_HIGH'
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'YRL_LOW'
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'P_E_RATIO'
    Width = 64
    Visible = True
  end
  item
    Expanded = False
    FieldName = 'PROJ_GRTH'
```

```
  Width = 281
  Height = 265
  BevelInner = bvRaised
  BevelOuter = bvLowered
  TabOrder = 0
  <------->object ProgramIcon:
TImage
    Left = 8
    Top = 8
    Width = 65
    Height = 57
    Picture.Data =000
      0000}
    Stretch = True
    IsControl = True
  end
  <------->object ProductName:
TLabel
    Left = 88
    Top = 16
    Width = 92
    Height = 13
    Caption = 'DBCtrlGrid Example'
    IsControl = True
  end
  <------->object Version: TLabel
    Left = 88
    Top = 40
    Width = 53
    Height = 13
    Caption = 'Version 1.0'
    IsControl = True
  end
  <------->object Memo1: TMemo
    Left = 8
    Top = 72
    Width = 265
    Height = 181
    Lines.Strings = (
      'This example illustrates'
      ''
      '-    The use of the DBCtrlGrid'
      '-    Calculating and Displaying
Total Fields'
      '-    Formatting of fields using
the format spec.'
      '-    Using Bookmarks'
      '-    Defining an Event Handler
```

```
  <------->object DBNavigator1:
TDBNavigator
    Left = 4
    Top = 168
    Width = 210
    Height = 18
    TabOrder = 3
  end
  <------->object MainMenu1:
TMainMenu
    Left = 165
    Top = 224
    <------->object About1:
TMenuItem
      Caption = '&About'
      OnClick = About1Click
    end
  end
End
```

الملف النصي لAboutBox

```
<------->object FMAboutBox:
TFMAboutBox
  Left = 189
  Top = 185
  BorderStyle = bsDialog
  Caption = 'About'
  ClientHeight = 321
  ClientWidth = 298
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <------->object Panel1: TPanel
    Left = 8
    Top = 8
```

tblMasterSYMBOL: TStringField
Alignment = taCenter
DisplayLabel = 'Symbol'
DisplayWidth = 7
FieldName = 'SYMBOL'
Size = 7
end
<------->object tblMasterCO_NAME: TStringField
DisplayLabel = 'Company'
DisplayWidth = 20
FieldName = 'CO_NAME'
end
<------->object tblMasterEXCHANGE: TStringField
DisplayLabel = 'Exchange'
DisplayWidth = 8
FieldName = 'EXCHANGE'
Size = 8
end
<------->object tblMasterIndustryLongName: TStringField
Alignment = taCenter
DisplayLabel = 'Industry'
DisplayWidth = 18
FieldKind = fkLookup
FieldName = 'IndustryLongName'
LookupDataSet = tblIndustry
LookupKeyFields = 'IND_CODE'
LookupResultField = 'LONG_NAME'
KeyFields = 'INDUSTRY'
Size = 32
Lookup = True
end
<------->object tblMasterCUR_PRICE: TFloatField
DisplayLabel = 'Current Price'
DisplayWidth = 10
FieldName = 'CUR_PRICE'
DisplayFormat = '$ #,###.00'
EditFormat = '#,###.00'
end
<------->object

at runtime'
''
'CtrlGrid uses the Stock, Holdings and Industry '
'tables from the DBDemos data to display a '
'listing of portfolio investments and their current '
'valuation.'
'')
ReadOnly = True
ScrollBars = ssVertical
TabOrder = 0
end
end
<------->object OKButton: TButton
Left = 103
Top = 284
Width = 77
Height = 27
Caption = 'OK'
Default = True
ModalResult = 1
TabOrder = 1
end
End
الملف النصي للوحدة dm1
<------->object DM1: TDM1
OldCreateOrder = True
OnCreate = DataModuleCreate
Left = 315
Top = 359
Height = 179
Width = 202
<------->object tblMaster: TTable
Active = True
DatabaseName = 'DBDEMOS'
IndexName = 'SYMBOL'
TableName = 'MASTER.DBF'
Left = 24
Top = 12
<------->object

DisplayFormat = '###%'
EditFormat = '###'
end
<------->object tblMasterRATING: TStringField
Alignment = taCenter
DisplayLabel = 'Rating'
DisplayWidth = 5
FieldName = 'RATING'
Size = 4
end
<------->object tblMasterRANK: TFloatField
Alignment = taCenter
DisplayLabel = 'Rank'
DisplayWidth = 5
FieldName = 'RANK'
end
<------->object tblMasterOUTLOOK: TSmallintField
Alignment = taCenter
DisplayLabel = 'Outlook'
DisplayWidth = 7
FieldName = 'OUTLOOK'
end
<------->object tblMasterRCMNDATION: TStringField
Alignment = taCenter
DisplayLabel = 'Recc.'
DisplayWidth = 6
FieldName = 'RCMNDATION'
Size = 5
end
<------->object tblMasterRISK: TStringField
Alignment = taCenter
DisplayLabel = 'Risk'
DisplayWidth = 6
FieldName = 'RISK'
Size = 4
end
end
<------->object dsMaster: TDataSource
DataSet = tblMaster
Left = 76

tblMasterYRL_HIGH: TFloatField
DisplayLabel = 'Yr. High'
DisplayWidth = 10
FieldName = 'YRL_HIGH'
DisplayFormat = '$ #,###.00'
EditFormat = '#,###.00'
end
<------->object tblMasterYRL_LOW: TFloatField
DisplayLabel = 'Yr. Low'
DisplayWidth = 10
FieldName = 'YRL_LOW'
DisplayFormat = '$ #,###.00'
EditFormat = '#,###.00'
end
<------->object tblMasterP_E_RATIO: TFloatField
Alignment = taCenter
DisplayLabel = 'P/E Ratio'
DisplayWidth = 10
FieldName = 'P_E_RATIO'
end
<------->object tblMasterPROJ_GRTH: TFloatField
Alignment = taCenter
DisplayLabel = 'Proj. Growth'
DisplayWidth = 10
FieldName = 'PROJ_GRTH'
DisplayFormat = '###%'
EditFormat = '###'
end
<------->object tblMasterINDUSTRY: TSmallintField
Alignment = taCenter
DisplayLabel = 'Industry'
DisplayWidth = 10
FieldName = 'INDUSTRY'
Visible = False
end
<------->object tblMasterPRICE_CHG: TSmallintField
Alignment = taCenter
DisplayLabel = 'Price Change'
DisplayWidth = 10
FieldName = 'PRICE_CHG'

```
   <------->object
tblHoldingsACCT_NBR:
TFloatField
      DisplayLabel = 'Account No.'
      FieldName = 'ACCT_NBR'
    end
   <------->object
tblHoldingsSHARES: TFloatField
      DisplayLabel = 'Shares'
      FieldName = 'SHARES'
      DisplayFormat = '###,###,###'
      EditFormat = '###,###,###'
    end
   <------->object
tblHoldingsPUR_PRICE:
TFloatField
      DisplayLabel = 'Purchase Price'
      FieldName = 'PUR_PRICE'
      DisplayFormat = '$ #,###.00'
      EditFormat = '#,###.00'
    end
   <------->object
tblHoldingsPUR_DATE: TDateField
      DisplayLabel = 'Purchase Date'
      FieldName = 'PUR_DATE'
      DisplayFormat = 'mmm. dd,
yyyy'
    end
   <------->object
tblHoldingsSYMBOL: TStringField
      FieldName = 'SYMBOL'
      Visible = False
      Size = 7
    end
   <------->object
tblHoldingsPUR_COST:
TCurrencyField
      FieldKind = fkCalculated
      FieldName = 'PUR_COST'
      Calculated = True
    end
  end
  <------->object dsHoldings:
TDataSource
    DataSet = tblHoldings
    Left = 80
    Top = 68
  end
End
    Top = 12
  end
  <------->object tblIndustry: TTable
    Active = True
    DatabaseName = 'DBDEMOS'
    IndexName = 'IND_CODE'
    MasterFields = 'INDUSTRY'
    TableName = 'INDUSTRY.DBF'
    Left = 140
    Top = 12
    <------->object
tblIndustryIND_CODE:
TSmallintField
      FieldName = 'IND_CODE'
      Visible = False
    end
    <------->object
tblIndustryIND_NAME:
TStringField
      FieldName = 'IND_NAME'
      Size = 5
    end
    <------->object
tblIndustryLONG_NAME:
TStringField
      FieldName = 'LONG_NAME'
    end
  end
  <------->object dsIndustry:
TDataSource
    DataSet = tblIndustry
    Left = 136
    Top = 68
  end
  <------->object tblHoldings: TTable
    Active = True
    AfterOpen =
tblHoldingsAfterOpen
    AfterPost = tblHoldingsAfterPost
    OnCalcFields =
tblHoldingsCalcFields
    DatabaseName = 'DBDEMOS'
    IndexName = 'SYMBOL'
    MasterFields = 'SYMBOL'
    MasterSource = dsMaster
    TableName = 'HOLDINGS.DBF'
    Left = 12
    Top = 68
```

انتهى البرنامج والآن إلى الصور المساعدة

4FishFact الملف التنفيذي للبرنامج :

FishFact

```
unit Ffactwin;

{ This application shows how to display Paradox style memo and graphic
 fields in a form. Table1's DatabaseName property should point to the
 Delphi sample database. Table1's TableName property should be set to
 the BIOLIFE table. }

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls,
  Forms, StdCtrls, DBCtrls, DBGrids, DB, DBTables, Buttons, Grids, ExtCtrls;

type
  TForm1 = class(TForm)
    Panel1: TPanel;
    Label1: TLabel;
    DBImage1: TDBImage;
    DBLabel1: TDBText;
    DBMemo1: TDBMemo;
    DataSource1: TDataSource;
    Table1: TTable;
    Table1Common_Name: TStringField;
    Table1Graphic: TBlobField;
    DBGrid1: TDBGrid;
    BitBtn1: TBitBtn;
    Table1Category: TStringField;
    Table1SpeciesName: TStringField;
    Table1Lengthcm: TFloatField;
    Table1Length_In: TFloatField;
    Table1Notes: TMemoField;
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.FormCreate(Sender: TObject);
begin

end;

end.
```

Left = 6
Top = 8
Width = 259
Height = 217
Hint = 'Scroll grid below to see other fish'
ParentShowHint = False
ShowHint = True
TabOrder = 0
<<-------object---- DBLabel1: TDBText
Left = 4
Top = 183
Width = 249
Height = 24
DataField = 'Common_Name'
DataSource = DataSource1
Font.Charset = DEFAULT_CHARSET
Font.Color = clRed
Font.Height = -19
Font.Name = 'MS Serif'
Font.Style = [fsBold, fsItalic]
ParentFont = False
end
<<-------object---- DBImage1: TDBImage
Left = 5
Top = 8
Width = 248
Height = 168
Hint = 'Scroll grid below to see other fish'
DataField = 'Graphic'
DataSource = DataSource1
TabOrder = 0
end
end
object Panel2: TPanel
Left = 275
Top = 8
Width = 223
Height = 22
TabOrder = 1
<<-------object---- Label1: TLabel
Left = 7
Top = 4
Width = 56

؛FishFact

4FishFact

program Fishfact;
إعداد علاء الدين اللباد ٠٩٤٤٥٧٥٣٧١ الملف النصي
Uses
Forms,
Ffactwin in 'FFACTWIN.PAS' {Form1};
{$R *.RES}
begin
Application.Initialize;
Application.CreateForm(TForm1, Form1);
Application.Run;
end.
object Form1: TForm1
Left = 223
Top = 158
ActiveControl = DBImage1
BorderIcons = [biSystemMenu, biMinimize]
BorderStyle = bsSingle
Caption = 'FISH FACTS'
ClientHeight = 394
ClientWidth = 596
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
ShowHint = True
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
object Panel1: TPanel

Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentCtl3D = False
ParentFont = False
ScrollBars = ssVertical
TabOrder = 0
end
end
object Panel4: TPanel
Left = 0
Top = 322
Width = 596
Height = 72
Align = alBottom
BevelInner = bvRaised
BorderStyle = bsSingle
ParentShowHint = False
ShowHint = True
TabOrder = 3
<<-------object---- DBGrid1: TDBGrid
Left = 12
Top = 8
Width = 386
Height = 53
Hint = 'Scroll up/down to see other fish!'
DataSource = DataSource1
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 0
TitleFont.Charset = DEFAULT_CHARSET
TitleFont.Color = clBlack
TitleFont.Height = -11
TitleFont.Name = 'MS Sans Serif'
TitleFont.Style = []
end
<<-------object---- BitBtn1: TBitBtn
Left = 424

Height = 13
Caption = 'About the'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
end
<<-------object---- DBLabel2: TDBText
Left = 67
Top = 4
Width = 99
Height = 13
AutoSize = True
DataField = 'Common_Name'
DataSource = DataSource1
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
end
end
object Panel3: TPanel
Left = 276
Top = 32
Width = 223
Height = 193
BevelOuter = bvLowered
TabOrder = 2
<<-------object---- DBMemo1: TDBMemo
Left = 3
Top = 2
Width = 217
Height = 183
BorderStyle = bsNone
Color = clSilver
Ctl3D = False
DataField = 'Notes'
DataSource = DataSource1
Font.Charset = DEFAULT_CHARSET

Top = 20
Width = 57
Height = 29
Hint = 'Close Fish Facts'
Caption = 'E&xit'
TabOrder = 1
Kind = bkClose
end
end
object DataSource1: TDataSource
DataSet = Table1
Left = 19
Top = 193
end
object Table1: TTable
Active = True
DatabaseName = 'DBDEMOS'
ReadOnly = True
TableName = 'BIOLIFE'
Left = 221
Top = 193
<<-------object---- Table1Category: TStringField
DisplayWidth = 15
FieldName = 'Category'
Size = 15
end
<<-------object---- Table1SpeciesName: TStringField
DisplayWidth = 20
FieldName = 'Species Name'
Size = 40
end
<<-------object---- Table1Lengthcm: TFloatField
DisplayWidth = 11
FieldName = 'Length (cm)'
end
<<-------object---- Table1Length_In: TFloatField
DisplayWidth = 10
FieldName = 'Length_In'
DisplayFormat = '0.00'
end
<<-------object---- Table1Common_Name: TStringField
DisplayWidth = 30

FieldName = 'Common_Name'
Visible = False
Size = 30
end
<<-------object---- Table1Notes: TMemoField
FieldName = 'Notes'
BlobType = ftMemo
Size = 50
end
<<-------object---- Table1Graphic: TBlobField
FieldName = 'Graphic'
ReadOnly = True
Visible = False
end
end
End

الهدف من هذا البرنامج هو تعريف الطالب بكيفية التعامل مع البيانات المخزنة ضمن data base وذلك بإدخالها ضمن برنامج الدلفي من اجل حمايتها أولا ومن ثم إظهارها بالمظهر اللائق وتصميم الواجهة حسب ذوق المستثمر وإظهار الإبداع لدى المبرمج من اجل إيصال الفكرة المقصودة من البرنامج و قد تم شرح هذا البرنامج خطوة خطوة ويجب الانتباه إلى ما يلي في أي قاعدة بيانات مستوردة :

١- في التعليمة Table1 يجب ان يكون :

object Table1: TTable
Active = True
DatabaseName = 'DBDEMOS Name=table1'
ReadOnly = True
TableName = 'BIOLIFE'
Left = 221
Top = 193

ثم ننقر بزر الفارة الأيمن على التعليمة table1 ثم نختار الحقول المطلوبة وننقر add أي اضافة الحقول الى الجدول

٢- في التعليمة DataSource1 يجب ان يكون :

object DataSource1: TDataSource
DataSet = Table1 Tablename=biolife.db Active=true Name = DataSource1 Databasename=DBDemos
Left = 19
Top = 193
end

٣- في التعليمة Dbgrid1 يجب أن يكون :

-----object---- DBGrid1: TDBGrid
Left = 12
Top = 8
Width = 386
Height = 53
Hint = 'Scroll up/down to see other fish!'
DataSource = DataSource1 Dataset = table1 Active = true Databasename=DBDemos Tablename=biolife.db
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 0
TitleFont.Charset = DEFAULT_CHARSET
TitleFont.Color = clBlack
TitleFont.Height = -11
TitleFont.Name = 'MS Sans Serif'
TitleFont.Style = []
end

٤- يجب ان يتم التعامل مع التعليمات السابقة حسب التسلسل أي نبدأ من ١ الى ٣

وبالنسبة للأزرار الموجودة على الشكل فهي موزعة على الصفحات كما يلي :

object Form1: TForm1:
standard
DBLabel1 : datacontrol

dataAccess,Bde,Additional
أي ان التعامل مع ٥ صفحات فقط

unit bmpformu;
// A TBitmap is an encapsulation of the windows BITMAP and PALETTE which
// manages realizing of the palette automatically.
//
// The bitmap can be loaded from a .BMP file (LoadFromFile method) or a
// resource (LoadFromResourceName or LoadFromResourceID method) and saved
// back to a file (SaveToFile method). It can be drawn on a canvas by
// using the TCanvas' Draw or StretchDraw method. The size of the bitmap
// can be determined by using the Height and Width properties of TBitmap.
//
// The example below illustrates the use of the following:
//
// TBitmap, Palette, LoadFromFile, Draw, Height, Width
// Be sure that a small bitmap file named bor6.bmp is present in the same
// directory as the .exe file.
// Disclaimer: This is an example of using TBitmaps and is not intended to
// be an efficient method of tiling a form.
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls;

ObjectS
الملف الكامل النصي والتنفيذي لبرنامج
(6Bitmap)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١
برنامج لتغيير لون الخلفية
إ
الملف التنفيذي
Procedures 12

```
type
  TBmpForm = class(TForm)
    Button1: TButton;
    <-------> procedure
FormDestroy(Sender: TObject);
    <-------> procedure
FormPaint(Sender: TObject);
    <-------> procedure
Button1Click(Sender: TObject);
    <-------> procedure
FormCreate(Sender: TObject);
  private
    { Private declarations }
    Bitmap: TBitmap;
    <-------> procedure
ScrambleBitmap;
    <-------> procedure
WMEraseBkgnd(var m:
TWMEraseBkgnd); message
WM_ERASEBKGND;
  Public
    { Public declarations }
  end;

Var
  BmpForm: TBmpForm;

Implementation


{$R *.DFM}


<-------> procedure
TBmpForm.FormCreate(Sender:
TObject);
begin
  Bitmap := TBitmap.Create;

Bitmap.LoadFromFile('bor6.bmp');
end;

<-------> procedure
TBmpForm.FormDestroy(Sender:
TObject);
begin
  Bitmap.Free;
end;

```

```
// since we're going to be painting the
whole form, handling this
// message will suppress the
uneccessary repainting of the
background
// which can result in flicker.
<-------> procedure
TBmpform.WMEraseBkgnd(var m :
TWMEraseBkgnd);
begin
  m.Result := LRESULT(False);
end;

<-------> procedure
TBmpForm.FormPaint(Sender:
TObject);
var
  x, y: Integer;
begin
  y := 0;
  while y < Height do
  begin
    x := 0;
    while x < Width do
    begin
      Canvas.Draw(x, y, Bitmap);
      x := x + Bitmap.Width;
    end;
    y := y + Bitmap.Height;
  end;
end;

<-------> procedure
TBmpForm.Button1Click(Sender:
TObject);
begin
  ScrambleBitmap;
  Invalidate;
end;


//
// scrambling the bitmap is easy
when it's has 256 colors:
// we just need to change each of the
color in the palette
// to some other value.
//
<-------> procedure
```

```
TBmpForm.ScrambleBitmap;
var
  pal: PLogPalette;
  hpal: HPALETTE;
  i: Integer;
begin
  pal := nil;
  try
    GetMem(pal, sizeof(TLogPalette) + sizeof(TPaletteEntry) * 255);
    pal.palVersion := $300;
    pal.palNumEntries := 256;
    for i := 0 to 255 do
    begin
      pal.palPalEntry[i].peRed := Random(255);
      pal.palPalEntry[i].peGreen := Random(255);
      pal.palPalEntry[i].peBlue := Random(255);
    end;
    hpal := CreatePalette(pal^);
    if hpal <> 0 then
      Bitmap.Palette := hpal;
  finally
    FreeMem(pal);
  end;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

الملف النصي للبرنامج

```
object BmpForm: TBmpForm
  Left = 136
  Top = 124
  Width = 466
  Height = 334
  Caption = 'BmpForm'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  OnCreate = FormCreate
  OnDestroy = FormDestroy
  OnPaint = FormPaint
  PixelsPerInch = 96
  TextHeight = 13
  object Button1: TButton
    Left = 56
    Top = 40
    Width = 113
    Height = 25
    Caption = 'Scramble Palette'
    TabOrder = 0
    OnClick = Button1Click
  end
End
```

ObjectS

الملف التنفيذي لبرنامج

(Combobox)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit combform;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls, ComCtrls, Menus,
ExtCtrls;

type
  TComboForm = class(TForm)
    CbDrop: TComboBox;
    CbDropList: TComboBox;
    CbSimple: TComboBox;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    MainMenu1: TMainMenu;
    ComboStyle: TMenuItem;
    CmSimple: TMenuItem;
    CmDrop: TMenuItem;
    CmDropList: TMenuItem;
    Add1: TMenuItem;
    CmAddString: TMenuItem;
    CmAddStringAt: TMenuItem;
    Search1: TMenuItem;
    CmFindString: TMenuItem;
    CmFindIndex: TMenuItem;
    CmDelete: TMenuItem;
    CmList: TMenuItem;
    CmDelString: TMenuItem;
    CmDelIndex: TMenuItem;
    CmShowList: TMenuItem;
    CmClear: TMenuItem;
    N1: TMenuItem;
    Label5: TLabel;
    EdCursel: TEdit;
    EdCuridx: TEdit;
    EdCurlen: TEdit;
    EdEdit: TEdit;
    EdEditlen: TEdit;
    CmSortList: TMenuItem;
    Bevel1: TBevel;
    procedure
ComboStyleClick(Sender: TObject);
    procedure
CmSimpleClick(Sender: TObject);
    procedure CmDropClick(Sender:
TObject);
    procedure
```

```
  property CbActive: TComboBox read FCbActive write SetCbActive;
 public
  { Public declarations }
 end;

var
 ComboForm: TComboForm;

implementation

uses inptform;

{$R *.DFM}

// -------------- //
// event handlers //
// -------------- //

procedure TComboForm.FormCreate(Sender: TObject);
begin
 FCbActive := CbSimple;
 CbSimple.Visible := True;
 CbDrop.Visible := False;
 CbDropList.Visible := False;
 NewSelection;
end;

// Control notifications
// ---------------------

//
// close form
//
procedure TComboForm.Button1Click(Sender: TObject);
begin
 Close;
end;

procedure TComboForm.ComboChange(Sender: TObject);
begin
```

```
CmDropListClick(Sender: TObject);
  procedure FormCreate(Sender: TObject);
  procedure ComboChange(Sender: TObject);
  procedure CmClearClick(Sender: TObject);
  procedure CmDelStringClick(Sender: TObject);
  procedure CmDelIndexClick(Sender: TObject);
  procedure CmFindStringClick(Sender: TObject);
  procedure CmFindIndexClick(Sender: TObject);
  procedure CmAddStringClick(Sender: TObject);
  procedure CmAddStringAtClick(Sender: TObject);
  procedure Button1Click(Sender: TObject);
  procedure CmSortListClick(Sender: TObject);
  procedure CmListClick(Sender: TObject);
  procedure CmShowListClick(Sender: TObject);
 private
  { Private declarations }
  FCbActive: TComboBox;
  procedure SetCbActive(cb: TComboBox);

  procedure NewSelection;
  procedure DeleteAt(idx: Integer);
  procedure AddAt(idx: Integer; s: string);
  procedure SelectAt(idx: Integer);

  function FindString(s: string): Integer;
```

```
add:', s) then
    AddAt(0, s);
end;

procedure
TComboForm.CmAddStringAtClick
(Sender: TObject);
var
  i: Integer;
  s: string;
begin
  if InputForm.GetString('String to
add:', s)
      and
InputForm.GetInteger('Index to
insert at:', i) then
    AddAt(i, s);
end;

// Find menu
// ---------
procedure
TComboForm.CmFindStringClick(S
ender: TObject);
var
  s: string;
begin
  if InputForm.GetString('String to
select:', s) then
    SelectAt(FindString(s));
end;

procedure
TComboForm.CmFindIndexClick(S
ender: TObject);
var
  i: Integer;
begin
  if InputForm.GetInteger('Index to
select:', i) then
    SelectAt(i);
end;


// Delete menu
// -----------
procedure
TComboForm.CmDelStringClick(Se
nder: TObject);
```

```
  NewSelection;
end;


// ComboBox menu
// -------------

procedure
TComboForm.ComboStyleClick(Sen
der: TObject);
begin
  CmSimple.Checked := CbActive =
CbSimple;
  CmDrop.Checked := CbActive =
CbDrop;
  CmDropList.Checked := CbActive
= CbDropList;
end;

procedure
TComboForm.CmSimpleClick(Send
er: TObject);
begin
  CbActive := CbSimple;
end;

procedure
TComboForm.CmDropClick(Sender
: TObject);
begin
  CbActive := CbDrop;
end;

procedure
TComboForm.CmDropListClick(Se
nder: TObject);
begin
  CbActive := CbDropList;
end;

// Add menu
// --------
procedure
TComboForm.CmAddStringClick(S
ender: TObject);
var
  s: string;
begin
  if InputForm.GetString('String to
```

```
  CmShowList.Enabled := CbActive <> CbSimple;
  CmSortList.Checked := CbActive.Sorted;
end;


// --------------- //
// private methods //
// --------------- //

procedure TComboForm.NewSelection;
begin
  if CbActive.ItemIndex = -1 then
    EdCursel.Text := ''
  else
    EdCursel.Text := CbActive.Items[CbActive.ItemIndex];
  EdCuridx.Text := IntToStr(CbActive.ItemIndex);
  EdCurlen.Text := IntToStr(length(EdCursel.Text));

  EdEdit.Text := CbActive.Text;
  EdEditlen.Text := IntToStr(length(EdEdit.Text));
end;

procedure TComboForm.SetCbActive(cb: TComboBox);
begin
  if FCbActive <> cb then
  begin
    FCbActive.Visible := False;
    FCbActive := cb;
    FCbActive.Visible := True;
    NewSelection;
  end;
end;


//
// Delete an item from the active combobox given its index.
// Ignores bogus indices.
//
```

```
var
  s: string;
begin
  if InputForm.GetString('String to delete:', s) then
    DeleteAt(FindString(s));
end;

procedure TComboForm.CmDelIndexClick(Sender: TObject);
var
  i: Integer;
begin
  if InputForm.GetInteger('Index to delete:', i) then
    DeleteAt(i);
end;

procedure TComboForm.CmClearClick(Sender: TObject);
begin
  CbActive.Clear;
end;


// List menu
// ---------
procedure TComboForm.CmShowListClick(Sender: TObject);
begin
  CbActive.DroppedDown := not CbActive.DroppedDown;
end;

procedure TComboForm.CmSortListClick(Sender: TObject);
begin
  CbActive.Sorted := not CbActive.Sorted;
end;

procedure TComboForm.CmListClick(Sender: TObject);
begin
```

1.
//
function TComboForm.FindString(s: string): Integer;
var
i: Integer;
begin
i := 0;
while (i < CbActive.Items.Count) and (CbActive.Items[i] <> s) do
inc(i);
Result := i;
end;
end.

Object$
الملف النصي لبرنامج
(Combobox)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١
object ComboForm: TComboForm
Left = 227
Top = 141
BorderIcons = [biSystemMenu, biMinimize]
BorderStyle = bsSingle
Caption = 'ComboBox demo'
ClientHeight = 281
ClientWidth = 611
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
Menu = MainMenu1

procedure TComboForm.DeleteAt(idx: Integer);
begin
if idx >= 0 then
CbActive.Items.Delete(idx);
end;
//
// Add a string into the active combobox at the given index.
// If the index is too big, append the item, if too small,
// insert it at the top.
//
procedure TComboForm.AddAt(idx: Integer; s: string);
begin
if idx > CbActive.Items.Count then
CbActive.Items.Append(s)
else
begin
if idx < 0 then
idx := 0;
CbActive.Items.Insert(idx, s);
end;
end;
//
// Select an item in the active combobox given it's index.
// A bogus index will result in an exception.
procedure TComboForm.SelectAt(idx: Integer);
begin
CbActive.ItemIndex := idx;
NewSelection;
end;
//
// Find the index of an item in the combobox given a string.
// If the string does not exist, return -

```
  OldCreateOrder = True
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  object Label1: TLabel
    Left = 288
    Top = 48
    Width = 82
    Height = 13
    Caption = 'Current selection:'
  end
  object Label2: TLabel
    Left = 305
    Top = 82
    Width = 65
    Height = 13
    Caption = 'Current index:'
  end
  object Label3: TLabel
    Left = 241
    Top = 116
    Width = 129
    Height = 13
    Caption = 'Length of current
selection:'
  end
  object Label4: TLabel
    Left = 321
    Top = 150
    Width = 49
    Height = 13
    Caption = 'Edit string:'
  end
  object Label5: TLabel
    Left = 271
    Top = 176
    Width = 99
    Height = 13
    Caption = 'Length of Edit String:'
  end
  object Bevel1: TBevel
    Left = 0
    Top = 0
    Width = 611
    Height = 2
    Align = alTop
  end
  object CbDrop: TComboBox
    Left = 40
    Top = 48
    Width = 178
    Height = 21
    ItemHeight = 13
    TabOrder = 0
    Visible = False
    OnChange = ComboChange
    Items.Strings = (
      'in'
      'this'
      'box'
      'we'
      'have'
      'a'
      'list'
      'of'
      'strings')
  end
  object CbDropList: TComboBox
    Left = 40
    Top = 48
    Width = 178
    Height = 21
    Style = csDropDownList
    ItemHeight = 13
    TabOrder = 1
    Visible = False
    OnChange = ComboChange
    Items.Strings = (
      'Closer'
      'Still'
      'Substance'
      'Little Earthquakes'
      'Under the Pink'
      'Boys for Pele')
  end
  object CbSimple: TComboBox
    Left = 40
    Top = 48
    Width = 178
    Height = 129
    Style = csSimple
    ItemHeight = 13
    TabOrder = 2
    OnChange = ComboChange
    Items.Strings = (
      'Return of the Native'
```

```
  Top = 8
  object ComboStyle: TMenuItem
    Caption = '&ComboBox'
    OnClick = ComboStyleClick
    object CmSimple: TMenuItem
      Caption = '&Simple'
      OnClick = CmSimpleClick
    end
    object CmDrop: TMenuItem
      Caption = '&Dropdown'
      OnClick = CmDropClick
    end
    object CmDropList: TMenuItem
      Caption = 'Dropdown &list'
      OnClick = CmDropListClick
    end
  end
  object Add1: TMenuItem
    Caption = '&Add'
    object CmAddString:
TMenuItem
      Caption = '&String...'
      OnClick = CmAddStringClick
    end
    object CmAddStringAt:
TMenuItem
      Caption = 'String &at...'
      OnClick =
CmAddStringAtClick
    end
  end
  object Search1: TMenuItem
    Caption = '&Find'
    object CmFindString:
TMenuItem
      Caption = '&String...'
      OnClick = CmFindStringClick
    end
    object CmFindIndex:
TMenuItem
      Caption = '&Index...'
      OnClick = CmFindIndexClick
    end
  end
  object CmDelete: TMenuItem
    Caption = '&Delete'
    object CmDelString: TMenuItem
      Caption = '&String...'
```

```
    'Jude the Obscure'
    'One Hundred Years of Solitude'
    'The Autumn of the Patriarch'
    'Idoru'
    'Neuromancer')
end
object EdCursel: TEdit
  Left = 384
  Top = 48
  Width = 178
  Height = 21
  ReadOnly = True
  TabOrder = 3
end
object EdCuridx: TEdit
  Left = 384
  Top = 80
  Width = 178
  Height = 21
  ReadOnly = True
  TabOrder = 4
end
object EdCurlen: TEdit
  Left = 384
  Top = 112
  Width = 178
  Height = 21
  ReadOnly = True
  TabOrder = 5
end
object EdEdit: TEdit
  Left = 384
  Top = 144
  Width = 178
  Height = 21
  ReadOnly = True
  TabOrder = 6
end
object EdEditlen: TEdit
  Left = 384
  Top = 176
  Width = 178
  Height = 21
  ReadOnly = True
  TabOrder = 7
end
object MainMenu1: TMainMenu
  Left = 8
```

```
  PixelsPerInch = 96
  TextHeight = 13
  object PromptLabel: TLabel
    Left = 40
    Top = 8
    Width = 5
    Height = 13
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentFont = False
  end
  object InputEdit: TEdit
    Left = 36
    Top = 24
    Width = 193
    Height = 21
    TabOrder = 0
  end
  object Button1: TButton
    Left = 43
    Top = 72
    Width = 75
    Height = 25
    Caption = '&OK'
    Default = True
    ModalResult = 1
    TabOrder = 1
  end
  object Button2: TButton
    Left = 147
    Top = 72
    Width = 75
    Height = 25
    Caption = '&Cancel'
    ModalResult = 2
    TabOrder = 2
  end
End
```

```
        OnClick = CmDelStringClick
      end
      object CmDelIndex: TMenuItem
        Caption = '&Index...'
        OnClick = CmDelIndexClick
      end
      object N1: TMenuItem
        Caption = '-'
      end
      object CmClear: TMenuItem
        Caption = '&All'
        OnClick = CmClearClick
      end
    end
    object CmList: TMenuItem
      Caption = '&List'
      OnClick = CmListClick
      object CmShowList: TMenuItem
        Caption = '&Show'
        OnClick = CmShowListClick
      end
      object CmSortList: TMenuItem
        Caption = 'So&rt'
        OnClick = CmSortListClick
      end
    end
  end

End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

object InputForm: TInputForm
  Left = 150
  Top = 147
  Width = 274
  Height = 140
  ActiveControl = InputEdit
  Caption = 'InputForm'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
```

```
  Forms,
  Main in 'Main.pas' {FmMain},
  DM1 in 'Dm1.pas' {DM},
  About in 'About.pas'
{FmAboutBox};

{$R *.RES}

begin
  Application.Initialize;

Application.CreateForm(TFmMain,
FmMain);
  Application.CreateForm(TDM,
DM);
  Application.Run;
end.



\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

unit Main;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  Grids, DBGrids, ExtCtrls, DBCtrls,
DB, DBTables, StdCtrls, Menus;

type
  TFmMain = class(TForm)
    DBNavigator1: TDBNavigator;
    GridCustomers: TDBGrid;
    GridOrders: TDBGrid;
    GridItems: TDBGrid;
    Label1: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    MainMenu1: TMainMenu;
    About1: TMenuItem;
    procedure
GridOrdersEnter(Sender: TObject);
    procedure
GridCustomersEnter(Sender:
```

ObjectS

الملف التنفيذي لبرنامج

(DbErrors)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
program DBErrors;

uses
```

```
er: TObject);
begin
  if Dm.Customer.State in
[dsEdit,dsInsert] then
    Dm.Customer.Post; {required if
user clicks onto details
                    after changing key so
that cascaded
                    update displays
properly}
end;

procedure
TFmMain.About1Click(Sender:
TObject);
var
  fmAboutBox : TFmAboutBox;
begin
  FmAboutBox :=
TFmAboutBox.Create(self);
  try
    FmAboutBox.showModal;
  finally
    FmAboutBox.free;
  end;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

(*
  This example represents a sampling
of the way that you might
  approach trapping a number of
database errors.


  A complete listing of the database
errorcodes is found in the
  DBIErrs.Int file in the Delphi/Doc
directory or in the IDAPI.h
  file in the Borland Database
Engine.

  Database errors are defined by
category and code. Here's a sample:

{ ERRCAT_INTEGRITY }
```

```
TObject);
    procedure
GridItemsEnter(Sender: TObject);
    procedure
GridCustomersExit(Sender:
TObject);
    procedure About1Click(Sender:
TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FmMain: TFmMain;

implementation

uses DM1, About;

  {$R *.dfm}

procedure
TFmMain.GridOrdersEnter(Sender
: TObject);
begin
  DBNavigator1.DataSource :=
Dm.OrdersSource;
end;

procedure
TFmMain.GridCustomersEnter(Sen
der: TObject);
begin
  DBNavigator1.DataSource :=
Dm.CustomerSource;
end;

procedure
TFmMain.GridItemsEnter(Sender:
TObject);
begin
  DBNavigator1.DataSource :=
Dm.ItemsSource;
end;

procedure
TFmMain.GridCustomersExit(Send
```

```
DBIERR_DETAILRECORDSEXIST = (ERRBASE_INTEGRITY + ERRCODE_DETAILRECORDSEXIST);
  DBIERR_FORIEGNKEYERR = (ERRBASE_INTEGRITY + ERRCODE_FORIEGNKEYERR);

  The ERRBASE_INTEGRITY value is $2600 (Hex 2600) or 9728 decimal.
  Thus, for example, the errorcode for keyviol is 9729
                              for master with details is 9734.

  *)

unit DM1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  DBTables, DB;

type
  TDM = class(TDataModule)
    Customer: TTable;
    CustomerCustNo: TFloatField;
    CustomerCompany: TStringField;
    CustomerSource: TDataSource;
    Orders: TTable;
    OrdersSource: TDataSource;
    Items: TTable;
    ItemsOrderNo: TFloatField;
    ItemsItemNo: TFloatField;
    ItemsPartNo: TFloatField;
    ItemsQty: TIntegerField;
    ItemsDiscount: TFloatField;
    ItemsSource: TDataSource;
    OrdersOrderNo: TFloatField;
    OrdersCustNo: TFloatField;
    OrdersSaleDate: TDateTimeField;
    OrdersShipDate: TDateTimeField;
```

```
  ERRCODE_KEYVIOL          = 1;  { Key violation }
  ERRCODE_MINVALERR        = 2;  { Min val check failed }
  ERRCODE_MAXVALERR        = 3;  { Max val check failed }
  ERRCODE_REQDERR          = 4;  { Field value required }
  ERRCODE_FORIEGNKEYERR    = 5;  { Master record missing }
  ERRCODE_DETAILRECORDSEXIST    = 6;  { Cannot MODIFY or DELETE this Master record }
  ERRCODE_MASTERTBLLEVEL   = 7;  { Master Table Level is incorrect }
  ERRCODE_LOOKUPTABLEERR   = 8;  { Field value out of lookup tbl range }
  ERRCODE_LOOKUPTBLOPENERR      = 9;  { Lookup Table Open failed }
  ERRCODE_DETAILTBLOPENERR      = 10; { 0x0a Detail Table Open failed }
  ERRCODE_MASTERTBLOPENERR      = 11; { 0x0b Master Table Open failed }
  ERRCODE_FIELDISBLANK     = 12; { 0x0c Field is blank }


  The constant for the base category is added to these constants to represent
  a unique DBI errorcode;

  DBIERR_KEYVIOL  = (ERRBASE_INTEGRITY + ERRCODE_KEYVIOL);
  DBIERR_REQDERR = (ERRBASE_INTEGRITY + ERRCODE_REQDERR);
```

```
  if (E is EDBEngineError) then
    if (E as EDBEngineError).Errors[0].Errorcode = eKeyViol then
    begin
      MessageDlg('Unable to post: Duplicate Customer ID.', mtWarning, [mbOK], 0);
      Abort;
    end;
 end;

procedure TDM.CustomerDeleteError(DataSet: TDataSet;
  E: EDatabaseError; var Action: TDataAction);
begin
  if (E is EDBEngineError) then
    if (E as EDBEngineError).Errors[0].Errorcode = eDetailsExist then
    {the customer record has dependent details in the Orders table.}
    begin
      MessageDlg('To delete this record, first delete related orders and items.',
        mtWarning, [mbOK], 0);
      Abort;
    end;
end;

procedure TDM.ItemsPostError(DataSet: TDataSet; E: EDatabaseError;
  var Action: TDataAction);
begin
  {This error will occur when a part number is specified that
   is not in the parts table.}
  if (E as EDBEngineError).Errors[0].Errorcode = eForeignKey then
  begin
    MessageDlg('Part number is invalid', mtWarning,[mbOK],0);
    Abort;
```

```
    OrdersEmpNo: TIntegerField;
    procedure CustomerPostError(DataSet: TDataSet; E: EDatabaseError;
      var Action: TDataAction);
    procedure CustomerDeleteError(DataSet: TDataSet; E: EDatabaseError;
      var Action: TDataAction);
    procedure ItemsPostError(DataSet: TDataSet; E: EDatabaseError;
      var Action: TDataAction);
    procedure OrdersPostError(DataSet: TDataSet; E: EDatabaseError;
      var Action: TDataAction);
    procedure OrdersDeleteError(DataSet: TDataSet; E: EDatabaseError;
      var Action: TDataAction);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  DM: TDM;

const
  {Declare constants we're interested in}
  eKeyViol = 9729;
  eRequiredFieldMissing = 9732;
  eForeignKey = 9733;
  eDetailsExist = 9734;


implementation

{$R *.dfm}

procedure TDM.CustomerPostError(DataSet: TDataSet;
  E: EDatabaseError; var Action: TDataAction);
begin
```

```
and related items?',
mtConfirmation,
      [mbYes, mbNo], 0) = mrYes
then
    begin
      {Delete related records in linked
'items' table}
      while Items.RecordCount > 0 do
        Items.delete;
      {Finally,delete this record}
      Action := daRetry;
    end else Abort;
  end;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

unit About;

interface

uses Windows, SysUtils, Classes,
Graphics, Forms, Controls, StdCtrls,
  Buttons, ExtCtrls, ComCtrls;

type
  TFmAboutBox = class(TForm)
    Panel1: TPanel;
    ProgramIcon: TImage;
    ProductName: TLabel;
    Version: TLabel;
    Panel2: TPanel;
    Label2: TLabel;
    Image1: TImage;
    Memo1: TMemo;
    Button1: TButton;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FmAboutBox: TFmAboutBox;

implementation
```

```
  end;
end;

procedure
TDM.OrdersPostError(DataSet:
TDataSet; E: EDatabaseError;
  var Action: TDataAction);
var
  iDBIError: Integer;
begin
  if (E is EDBEngineError) then
  begin
    iDBIError := (E as
EDBEngineError).Errors[0].Errorc
ode;
    case iDBIError of
      eRequiredFieldMissing:
        {The EmpNo field is defined as
being required.}
        begin
          MessageDlg('Please provide an
Employee ID', mtWarning, [mbOK],
0);
          Abort;
        end;
      eKeyViol:
        {The primary key is OrderNo}
        begin
          MessageDlg('Unable to post.
Duplicate Order Number',
mtWarning,
            [mbOK], 0);
          Abort;
        end;
    end;
  end;
end;

procedure
TDM.OrdersDeleteError(DataSet:
TDataSet; E: EDatabaseError;
  var Action: TDataAction);
begin
  if E is EDBEngineError then
    if (E as
EDBEngineError).Errors[0].Errorc
ode = eDetailsExist then
    begin
      if MessageDlg('Delete this order
```

```
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentFont = False
  end
  object Label3: TLabel
    Left = 8
    Top = 121
    Width = 48
    Height = 16
    Caption = 'Orders'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentFont = False
  end
  object Label4: TLabel
    Left = 8
    Top = 209
    Width = 38
    Height = 16
    Caption = 'Items'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentFont = False
  end
  object DBNavigator1:
TDBNavigator
    Left = 5
    Top = 4
    Width = 240
    Height = 25
    TabOrder = 0
  end
  object GridCustomers: TDBGrid
    Left = 8
    Top = 47
    Width = 342
    Height = 73
    DataSource =
DM.CustomerSource
```

```

{$R *.dfm}

end.



\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

Object$

الملف النصي لبرنامج

(DbErrors14)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
object FmMain: TFmMain
  Left = 280
  Top = 286
  BorderStyle = bsSingle
  Caption = 'Database Errors Demo'
  ClientHeight = 352
  ClientWidth = 359
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  Menu = MainMenu1
  OldCreateOrder = True
  PixelsPerInch = 96
  TextHeight = 13
  object Label1: TLabel
    Left = 8
    Top = 32
    Width = 74
    Height = 16
    Caption = 'Customers'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clWindowText
```

```
      FieldName = 'SaleDate'
      Visible = True
    end
    item
      Expanded = False
      FieldName = 'ShipDate'
      Visible = True
    end
    item
      Expanded = False
      FieldName = 'EmpNo'
      Width = 47
      Visible = True
    end>
  end
  object GridItems: TDBGrid
    Left = 8
    Top = 224
    Width = 341
    Height = 120
    DataSource = DM.ItemsSource
    TabOrder = 3
    TitleFont.Charset =
DEFAULT_CHARSET
    TitleFont.Color = clWindowText
    TitleFont.Height = -11
    TitleFont.Name = 'MS Sans Serif'
    TitleFont.Style = []
    OnEnter = GridItemsEnter
  end
  object MainMenu1: TMainMenu
    Left = 296
    object About1: TMenuItem
      Caption = '&About'
      OnClick = About1Click
    end
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

object FmAboutBox: TFmAboutBox
  Left = 224
  Top = 279
  BorderStyle = bsDialog
  Caption = 'About'
  ClientHeight = 403
  ClientWidth = 419
```

```
    TabOrder = 1
    TitleFont.Charset =
DEFAULT_CHARSET
    TitleFont.Color = clWindowText
    TitleFont.Height = -11
    TitleFont.Name = 'MS Sans Serif'
    TitleFont.Style = []
    OnEnter = GridCustomersEnter
    OnExit = GridCustomersExit
    Columns = <
      item
        Expanded = False
        FieldName = 'CustNo'
        Visible = True
      end
      item
        Expanded = False
        FieldName = 'Company'
        Visible = True
      end>
  end
  object GridOrders: TDBGrid
    Left = 8
    Top = 136
    Width = 341
    Height = 75
    DataSource = DM.OrdersSource
    TabOrder = 2
    TitleFont.Charset =
DEFAULT_CHARSET
    TitleFont.Color = clWindowText
    TitleFont.Height = -11
    TitleFont.Name = 'MS Sans Serif'
    TitleFont.Style = []
    OnEnter = GridOrdersEnter
    Columns = <
      item
        Expanded = False
        FieldName = 'OrderNo'
        Visible = True
      end
      item
        Expanded = False
        FieldName = 'CustNo'
        Visible = True
      end
      item
        Expanded = False
```

```
  object Panel2: TPanel
    Left = 9
    Top = 88
    Width = 401
    Height = 73
    TabOrder = 1
    object Label2: TLabel
      Left = 22
      Top = 8
      Width = 194
      Height = 13
      Caption = 'Data Model  -
Referential Integrity'
      Font.Charset =
DEFAULT_CHARSET
      Font.Color = clWindowText
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = [fsBold]
      ParentFont = False
    end
    object Image1: TImage
      Left = 18
      Top = 24
      Width = 325
      Height = 46
      Picture.Data = {
        }
    end
  end
  object Memo1: TMemo
    Left = 9
    Top = 168
    Width = 401
    Height = 193
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    Lines.Strings = (
      'This example demonstrates'
      ''
      '  - The use of Data Modules to
centralize coding'
      '  - A one-to-many-to-many form'
      '  - Trapping and Controlling
```

```
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  PixelsPerInch = 96
  TextHeight = 13
  object Panel1: TPanel
    Left = 8
    Top = 8
    Width = 402
    Height = 73
    BevelInner = bvRaised
    BevelOuter = bvLowered
    TabOrder = 0
    object ProgramIcon: TImage
      Left = 8
      Top = 8
      Width = 65
      Height = 57
      Picture.Data = {

        0000}
      Stretch = True
      IsControl = True
    end
    object ProductName: TLabel
      Left = 88
      Top = 16
      Width = 119
      Height = 13
      Caption = 'Database Errors
Example'
      IsControl = True
    end
    object Version: TLabel
      Left = 88
      Top = 40
      Width = 53
      Height = 13
      Caption = 'Version 1.0'
      IsControl = True
    end
  end
```

'General Notes:'
''
' The code in this example for the OnDeleteError on the Orders '
' table allows you to delete the current Orders record even if i' +
't '
' has dependent details. If this error occurs, a dialog box'
' asks you if you want to remove the Orders record and all'
' of its details. If this is action confirmed, the code deletes' +
' all'
' of the related Items records and then completes the action'
' by retrying the deletion for the current record. '
''
' The OnPostError, OnEditError and OnDeleteError pass to you'
' an exception (e) of the EDatabaseError class. To get '
' engine-level errorcodes, you must cast this as an error of'
' type EDBEngineError. See the code in DM.PAS for details.'
''
' The CustNo *Tfield* has min and max values. These are'
' NOT database engine-level Min and Max Valchecks and so'
' errors that arise from invalid entries are not caught by the '
' engine errors trapped for here.')
ParentFont = False
ReadOnly = True
ScrollBars = ssVertical
TabOrder = 2
WantReturns = False
end
object Button1: TButton
Left = 173
Top = 368
Width = 75

Database Errors'
''
'* Creation Order'
''
'The proper functioning of any form with a one-to-many '
'relationship depends on the creation order of TTables and '
'TDatasources at runtime. In this example, the MasterSource '
'property of the Orders table point to CustomerSource. For the '
'one-to-many link to properly synch up, the Customer table must '
'be active. The creation order is determined at design'
'time by right-clicking on the form and choosing Creation Order.'
''
'* Database Error Trapping'
''
'Delphi displays errors that arise from user interaction with '
'database tables as application level exceptions. The new '
'OnPostError, OnEditError and OnDeleteError events of '
'the TTable and TQuery objects allow you to trap these '
'errors closer to their origin. '
''
'To trigger these errors, run this application, and try to modify' +
','
'add, or delete records. '
''
'You may encounter several different database engine'
'errors: key violations, missing required fields, referential'
'integrity errors, incorrect foreign keys (most notably from '
'the "parts.db" table, which is not explicitly in this applicatio' +
'n), '
'and so on. '
''

```
    DataSet = Customer
    Left = 81
    Top = 8
  end
  object Orders: TTable
    Active = True
    OnDeleteError =
OrdersDeleteError
    OnPostError = OrdersPostError
    DatabaseName = 'DBDEMOS'
    IndexName = 'CustNo'
    MasterFields = 'CustNo'
    MasterSource = CustomerSource
    TableName = 'ORDERS.DB'
    Left = 17
    Top = 70
    object OrdersOrderNo:
TFloatField
      DisplayWidth = 10
      FieldName = 'OrderNo'
      DisplayFormat = '''#''0000'
      EditFormat = '0000'
    end
    object OrdersCustNo: TFloatField
      Alignment = taLeftJustify
      FieldName = 'CustNo'
      DisplayFormat = 'CN 0000'
      EditFormat = '0000'
      MaxValue =
9999.000000000000000000
      MinValue =
1000.000000000000000000
    end
    object OrdersSaleDate:
TDateTimeField
      FieldName = 'SaleDate'
    end
    object OrdersShipDate:
TDateTimeField
      FieldName = 'ShipDate'
    end
    object OrdersEmpNo:
TIntegerField
      FieldName = 'EmpNo'
      DisplayFormat = 'Emp''#'' 0000'
      EditFormat = '0000'
      MaxValue = 9999
      MinValue = 1
```

```
    Height = 25
    Cancel = True
    Caption = 'OK'
    Default = True
    ModalResult = 2
    TabOrder = 3
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

object DM: TDM
  OldCreateOrder = True
  Left = 347
  Top = 307
  Height = 242
  Width = 222
  object Customer: TTable
    Active = True
    OnDeleteError =
CustomerDeleteError
    OnPostError =
CustomerPostError
    DatabaseName = 'dbdemos'
    TableName = 'CUSTOMER.DB'
    Left = 17
    Top = 8
    object CustomerCustNo:
TFloatField
      Alignment = taLeftJustify
      DisplayWidth = 7
      FieldName = 'CustNo'
      DisplayFormat = 'CN 0000'
      EditFormat = '0000'
      MaxValue =
9999.000000000000000000
      MinValue =
1000.000000000000000000
    end
    object CustomerCompany:
TStringField
      DisplayWidth = 22
      FieldName = 'Company'
      Size = 30
    end
  end
  object CustomerSource:
TDataSource
```

```
   Top = 130
  end
End

\\\\\\
```

Procedures60

الملف التنفيذي لبرنامج

(Activefm)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
library EmpEditX;

{ This project demonstrates using
client datasets in an ActiveForm.
  The ActiveForm in this project
works just like the client in the
  EMPEDIT demo.  Before compiling
and using this Active Library you
  should compile and run the Server
project from the EMPEDIT demo.

  For infomation on deploying
ActiveForms, look for
  "ActiveX controls:deploying" in
the online help index.
}

uses
  ComServ,
  EmpE_TLB in 'EmpE_TLB.pas',
  EmpEdImp in 'EmpEdImp.pas'
{EmpEditForm: TActiveForm}
{EmpEditForm: CoClass},
  recerror in
'..\..\..\objrepos\recerror.pas' ;

{$E ocx}

exports
  DllGetClassObject,
  DllCanUnloadNow,
  DllRegisterServer,
```

```
   end
  end
  object OrdersSource: TDataSource
   DataSet = Orders
   Left = 81
   Top = 70
  end
  object Items: TTable
   Active = True
   OnPostError = ItemsPostError
   DatabaseName = 'DBDEMOS'
   IndexFieldNames = 'OrderNo'
   MasterFields = 'OrderNo'
   MasterSource = OrdersSource
   TableName = 'ITEMS.DB'
   Left = 17
   Top = 130
   object ItemsOrderNo: TFloatField
    DisplayWidth = 10
    FieldName = 'OrderNo'
    DisplayFormat = '0000'
   end
   object ItemsItemNo: TFloatField
    DisplayWidth = 10
    FieldName = 'ItemNo'
   end
   object ItemsPartNo: TFloatField
    Alignment = taLeftJustify
    DisplayWidth = 10
    FieldName = 'PartNo'
    DisplayFormat = 'PN-00000'
    EditFormat = '00000'
   end
   object ItemsQty: TIntegerField
    DisplayWidth = 6
    FieldName = 'Qty'
   end
   object ItemsDiscount: TFloatField
    DisplayWidth = 10
    FieldName = 'Discount'
    DisplayFormat = '#%'
    MinValue =
 100.000000000000000000
   end
  end
  object ItemsSource: TDataSource
   DataSet = Items
   Left = 81
```

UpdateButtonClick(Sender: TObject);
<------> procedure UndoButtonClick(Sender: TObject);
<------> procedure EmployeesReconcileError(DataSet: TCustomClientDataSet;
E: EReconcileError; UpdateKind: TUpdateKind; var Action: TReconcileAction);
<------> procedure EmpDataDataChange(Sender: TObject; Field: TField);
private
{ Private declarations }
FEvents: IEmpEditFormEvents;
<------> procedure ActivateEvent(Sender: TObject);
<------> procedure ClickEvent(Sender: TObject);
<------> procedure CreateEvent(Sender: TObject);
<------> procedure DblClickEvent(Sender: TObject);
<------> procedure DeactivateEvent(Sender: TObject);
<------> procedure DestroyEvent(Sender: TObject);
<------> procedure KeyPressEvent(Sender: TObject; var Key: Char);
<------> procedure PaintEvent(Sender: TObject);
protected
{ Protected declarations }
<------> procedure EventSinkChanged(const EventSink: IUnknown); override;
<------> procedure Initialize; override;
function Get_Active: WordBool; safecall;
function Get_AutoScroll: WordBool; safecall;
function Get_AxBorderStyle: TxActiveFormBorderStyle; safecall;
function Get_Caption: WideString; safecall;
function Get_Color: OLE_Color;

DllUnregisterServer;
{$R *.TLB}
{$R *.RES}
begin
end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
unit empedimp;
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
ActiveX, AxCtrls, EmpE_TLB, DBClient, Db, ComCtrls, ExtCtrls,
DBCtrls, StdCtrls, Mask, MConnect;
type
TEmpEditForm = class(TActiveForm, IEmpEditForm)
Label2: TLabel;
UpdateButton: TButton;
UndoButton: TButton;
QueryButton: TButton;
DBText1: TDBText;
FirstName: TDBEdit;
LastName: TDBEdit;
PhoneExt: TDBEdit;
HireDate: TDBEdit;
Salary: TDBEdit;
EmpData: TDataSource;
DBNavigator1: TDBNavigator;
Employees: TClientDataSet;
MidasConnection: TDCOMConnection;
RecInd: TLabel;
<------> procedure QueryButtonClick(Sender: TObject);
<------> procedure

```
   <------> procedure
Set_KeyPreview(Value: WordBool);
safecall;
   <------> procedure
Set_PixelsPerInch(Value: Integer);
safecall;
   <------> procedure
Set_PrintScale(Value:
TxPrintScale); safecall;
   <------> procedure
Set_Scaled(Value: WordBool);
safecall;
   <------> procedure
Set_Visible(Value: WordBool);
safecall;
   <------> procedure
Set_WindowState(Value:
TxWindowState); safecall;
  public
   { Public declarations }
  end;

implementation

uses RecError, ComServ;

{$R *.dfm}

{ TEmpEditForm }

<------> procedure
TEmpEditForm.EventSinkChanged(
const EventSink: IUnknown);
begin
  FEvents := EventSink as
IEmpEditFormEvents;
end;

<------> procedure
TEmpEditForm.Initialize;
begin
  OnActivate := ActivateEvent;
  OnClick := ClickEvent;
  OnCreate := CreateEvent;
  OnDblClick := DblClickEvent;
  OnDeactivate := DeactivateEvent;
  OnDestroy := DestroyEvent;
  OnKeyPress := KeyPressEvent;
  OnPaint := PaintEvent;
```

```
safecall;
   function Get_Cursor: Smallint;
safecall;
   function Get_DropTarget:
WordBool; safecall;
   function Get_Enabled: WordBool;
safecall;
   function Get_Font: Font; safecall;
   function Get_HelpFile:
WideString; safecall;
   function Get_KeyPreview:
WordBool; safecall;
   function Get_PixelsPerInch:
Integer; safecall;
   function Get_PrintScale:
TxPrintScale; safecall;
   function Get_Scaled: WordBool;
safecall;
   function Get_Visible: WordBool;
safecall;
   function Get_WindowState:
TxWindowState; safecall;
   <------> procedure
Set_AutoScroll(Value: WordBool);
safecall;
   <------> procedure
Set_AxBorderStyle(Value:
TxActiveFormBorderStyle);
safecall;
   <------> procedure
Set_Caption(const Value:
WideString); safecall;
   <------> procedure
Set_Color(Value: OLE_Color);
safecall;
   <------> procedure
Set_Cursor(Value: Smallint);
safecall;
   <------> procedure
Set_DropTarget(Value: WordBool);
safecall;
   <------> procedure
Set_Enabled(Value: WordBool);
safecall;
   <------> procedure Set_Font(const
Value: Font); safecall;
   <------> procedure
Set_HelpFile(const Value:
WideString); safecall;
```

```
end;

function TEmpEditForm.Get_Enabled: WordBool;
begin
  Result := Enabled;
end;

function TEmpEditForm.Get_Font: Font;
begin
  GetOleFont(Font, Result);
end;

function TEmpEditForm.Get_HelpFile: WideString;
begin
  Result := WideString(HelpFile);
end;

function TEmpEditForm.Get_KeyPreview: WordBool;
begin
  Result := KeyPreview;
end;

function TEmpEditForm.Get_PixelsPerInch: Integer;
begin
  Result := PixelsPerInch;
end;

function TEmpEditForm.Get_PrintScale: TxPrintScale;
begin
  Result := Ord(PrintScale);
end;

function TEmpEditForm.Get_Scaled: WordBool;
begin
  Result := Scaled;
end;
```

```
end;

function TEmpEditForm.Get_Active: WordBool;
begin
  Result := Active;
end;

function TEmpEditForm.Get_AutoScroll: WordBool;
begin
  Result := AutoScroll;
end;

function TEmpEditForm.Get_AxBorderStyle: TxActiveFormBorderStyle;
begin
  Result := Ord(AxBorderStyle);
end;

function TEmpEditForm.Get_Caption: WideString;
begin
  Result := WideString(Caption);
end;

function TEmpEditForm.Get_Color: OLE_COLOR;
begin
  Result := Color;
end;

function TEmpEditForm.Get_Cursor: Smallint;
begin
  Result := Smallint(Cursor);
end;

function TEmpEditForm.Get_DropTarget: WordBool;
begin
  Result := DropTarget;
```

```
end;

<------> procedure
TEmpEditForm.Set_DropTarget(Va
lue: WordBool);
begin
  DropTarget := Value;
end;

<------> procedure
TEmpEditForm.Set_Enabled(Value:
WordBool);
begin
  Enabled := Value;
end;

<------> procedure
TEmpEditForm.Set_Font(const
Value: Font);
begin
  SetOleFont(Font, Value);
end;

<------> procedure
TEmpEditForm.Set_HelpFile(const
Value: WideString);
begin
  HelpFile := String(Value);
end;

<------> procedure
TEmpEditForm.Set_KeyPreview(Va
lue: WordBool);
begin
  KeyPreview := Value;
end;

<------> procedure
TEmpEditForm.Set_PixelsPerInch(
Value: Integer);
begin
  PixelsPerInch := Value;
end;

<------> procedure
TEmpEditForm.Set_PrintScale(Val
ue: TxPrintScale);
begin
  PrintScale := TPrintScale(Value);
```

```

function
TEmpEditForm.Get_Visible:
WordBool;
begin
  Result := Visible;
end;

function
TEmpEditForm.Get_WindowState:
TxWindowState;
begin
  Result := Ord(WindowState);
end;

<------> procedure
TEmpEditForm.Set_AutoScroll(Val
ue: WordBool);
begin
  AutoScroll := Value;
end;

<------> procedure
TEmpEditForm.Set_AxBorderStyle(
Value: TxActiveFormBorderStyle);
begin
  AxBorderStyle :=
TActiveFormBorderStyle(Value);
end;

<------> procedure
TEmpEditForm.Set_Caption(const
Value: WideString);
begin
  Caption := TCaption(Value);
end;

<------> procedure
TEmpEditForm.Set_Color(Value:
OLE_COLOR);
begin
  Color := Value;
end;

<------> procedure
TEmpEditForm.Set_Cursor(Value:
Smallint);
begin
  Cursor := TCursor(Value);
```

```
TEmpEditForm.DblClickEvent(Sen
der: TObject);
begin
  if FEvents <> nil then
FEvents.OnDblClick;
end;

<------> procedure
TEmpEditForm.DeactivateEvent(Se
nder: TObject);
begin
  if FEvents <> nil then
FEvents.OnDeactivate;
end;

<------> procedure
TEmpEditForm.DestroyEvent(Send
er: TObject);
begin
  if FEvents <> nil then
FEvents.OnDestroy;
end;

<------> procedure
TEmpEditForm.KeyPressEvent(Sen
der: TObject; var Key: Char);
var
  TempKey: Smallint;
begin
  TempKey := Smallint(Key);
  if FEvents <> nil then
FEvents.OnKeyPress(TempKey);
  Key := Char(TempKey);
end;

<------> procedure
TEmpEditForm.PaintEvent(Sender:
TObject);
begin
  if FEvents <> nil then
FEvents.OnPaint;
end;

{
===========================
===========================
==================== }

<------> procedure
```

```
end;

<------> procedure
TEmpEditForm.Set_Scaled(Value:
WordBool);
begin
  Scaled := Value;
end;

<------> procedure
TEmpEditForm.Set_Visible(Value:
WordBool);
begin
  Visible := Value;
end;

<------> procedure
TEmpEditForm.Set_WindowState(
Value: TxWindowState);
begin
  WindowState :=
TWindowState(Value);
end;

<------> procedure
TEmpEditForm.ActivateEvent(Send
er: TObject);
begin
  if FEvents <> nil then
FEvents.OnActivate;
end;

<------> procedure
TEmpEditForm.ClickEvent(Sender:
TObject);
begin
  if FEvents <> nil then
FEvents.OnClick;
end;

<------> procedure
TEmpEditForm.CreateEvent(Sende
r: TObject);
begin
  if FEvents <> nil then
FEvents.OnCreate;
end;

<------> procedure
```

```
association with the change. }


Employees.UndoLastChange(True);

end;

<------> procedure
TEmpEditForm.EmployeesReconcileError(DataSet:
TCustomClientDataSet;
  E: EReconcileError; UpdateKind:
TUpdateKind; var Action:
TReconcileAction);
begin

{ This is the event handler which is
called when there are errors during
the
  update process.  To demonstrate,
you can create an error by running
two
  copies of this application and
modifying the same record in each
one.
  Here we use the standard reconcile
error dialog from the object
repository. }

  Action :=
HandleReconcileError(DataSet,
UpdateKind, E);
end;

<------> procedure
TEmpEditForm.EmpDataDataChange(Sender: TObject; Field: TField);
begin

{ This code is used to update the
status bar to show the number of
records
  that have been retrieved an our
relative position with the dataset. }

  with Employees do
    if Active then
      RecInd.Caption := Format(' %d
```

```
TEmpEditForm.QueryButtonClick(
Sender: TObject);
begin

{ Get data from the server.  The
number of records returned is
dependent on
  the PacketRecords property of
TClientDataSet.  If PacketRecords is
set
  to -1 then all records are returned.
Otherwise, as the user scrolls
  through the data, additional
records are returned in separate
data packets. }

  Employees.Close;
  Employees.Open;

end;

<------> procedure
TEmpEditForm.UpdateButtonClick
(Sender: TObject);
begin

{ Apply any edits.  The parameter
indicates the number of errors which
  are allowed before the updating is
aborted.  A value of -1 indicates that
  all successful updates be applied
regardless of the number of errors. }

  Employees.ApplyUpdates(-1);

end;

<------> procedure
TEmpEditForm.UndoButtonClick(S
ender: TObject);
begin

{ Here we demonstrate a new feature
in TClientDataSet, the ability to
undo
  changes in reverse order.  The
parameter indicates if the cursor
should
  be repositioned to the record
```

```
<------> Object EmpEditForm: TEmpEditForm
  Left = 219
  Top = 138
  Width = 453
  Height = 212
  Caption = 'Employee Administrator'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'Default'
  Font.Style = []
  OldCreateOrder = True
  PixelsPerInch = 96
  TextHeight = 13
  <------> Object Label2: TLabel
    Left = 28
    Top = 61
    Width = 50
    Height = 13
    Caption = 'First Name'
    FocusControl = FirstName
  end
  <------> Object Label3: TLabel
    Left = 123
    Top = 61
    Width = 61
    Height = 13
    AutoSize = False
    Caption = 'Last Name'
    FocusControl = LastName
  end
  <------> Object Label4: TLabel
    Left = 240
    Top = 62
    Width = 15
    Height = 13
    Caption = 'Ext'
    FocusControl = PhoneExt
  end
  <------> Object Label5: TLabel
    Left = 28
    Top = 118
```

```
of %d', [RecNo, RecordCount]);

end;
initialization
  TActiveFormFactory.Create(
    ComServer,
    TActiveFormControl,
    TEmpEditForm,
    Class_EmpEditForm,
    1,
    '',
    OLEMISC_SIMPLEFRAME or OLEMISC_ACTSLIKELABEL);
end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

<------> ObjectS22

الملف النصي لبرنامج

(Activefm)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
TDBEdit
  Left = 28
  Top = 76
  Width = 78
  Height = 21
  DataField = 'FirstName'
  DataSource = EmpData
  TabOrder = 0
 end
 <------> Object LastName:
TDBEdit
  Left = 123
  Top = 76
  Width = 99
  Height = 21
  DataField = 'LastName'
  DataSource = EmpData
  TabOrder = 1
 end
 <------> Object PhoneExt: TDBEdit
  Left = 240
  Top = 76
  Width = 49
  Height = 21
  DataField = 'PhoneExt'
  DataSource = EmpData
  TabOrder = 2
 end
 <------> Object HireDate: TDBEdit
  Left = 28
  Top = 132
  Width = 73
  Height = 21
  DataField = 'HireDate'
  DataSource = EmpData
  TabOrder = 3
 end
 <------> Object Salary: TDBEdit
  Left = 121
  Top = 132
  Width = 96
  Height = 21
  DataField = 'Salary'
  DataSource = EmpData
  TabOrder = 4
 end
 <------> Object QueryButton:
TButton
  Width = 45
  Height = 13
  Caption = 'Hire Date'
  FocusControl = HireDate
 end
 <------> Object Label10: TLabel
  Left = 123
  Top = 116
  Width = 29
  Height = 13
  Caption = 'Salary'
  FocusControl = Salary
 end
 <------> Object Label11: TLabel
  Left = 18
  Top = 18
  Width = 53
  Height = 16
  Alignment = taRightJustify
  Caption = 'Emp #: '
  Font.Charset = ANSI_CHARSET
  Font.Color = clBlack
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = [fsBold]
  ParentFont = False
 end
 <------> Object DBText1: TDBText
  Left = 73
  Top = 18
  Width = 68
  Height = 16
  DataField = 'EmpNo'
  DataSource = EmpData
  Font.Charset = ANSI_CHARSET
  Font.Color = clBlack
  Font.Height = -13
  Font.Name = 'MS Sans Serif'
  Font.Style = [fsBold]
  ParentFont = False
 end
 <------> Object RecInd: TLabel
  Left = 346
  Top = 16
  Width = 3
  Height = 13
 end
 <------> Object FirstName:
```

```
 <------> Object Employees: TClientDataSet
   Aggregates = <>
   PacketRecords = 10
   Params = <>
   ProviderName = 'EmpQueryProvider'
   RemoteServer = MidasConnection
   OnReconcileError = EmployeesReconcileError
   Left = 284
   Top = 12
 end
 <------> Object MidasConnection: TDCOMConnection
   ServerGUID = '{53BC6562-5B3E-11D0-9FFC-00A0248E4B9A}'
   ServerName = 'Serv.EmpServer'
   Left = 387
   Top = 12
 end
End
```

procedure s9

الملف لبرنامج

(----Briefcase)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
program BriefcaseDemo;

uses
  Forms,
  BriefcaseMain in 'BriefcaseMain.pas' {Form1};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TForm1,
```

```
   Left = 304
   Top = 88
   Width = 105
   Height = 25
   Caption = '&Get Employees'
   TabOrder = 5
   OnClick = QueryButtonClick
 end
 <------> Object DBNavigator1: TDBNavigator
   Left = 134
   Top = 14
   Width = 138
   Height = 25
   DataSource = EmpData
   VisibleButtons = [nbFirst, nbPrior, nbNext, nbLast, nbPost, nbCancel]
   TabOrder = 6
 end
 <------> Object UpdateButton: TButton
   Left = 344
   Top = 135
   Width = 105
   Height = 25
   Caption = '&Update Employees'
   TabOrder = 7
   OnClick = UpdateButtonClick
 end
 <------> Object UndoButton: TButton
   Left = 224
   Top = 134
   Width = 105
   Height = 25
   Caption = 'U&ndo Last Change'
   TabOrder = 8
   OnClick = UndoButtonClick
 end
 <------> Object EmpData: TDataSource
   DataSet = Employees
   OnDataChange = EmpDataDataChange
   Left = 331
   Top = 36
 end
```

```
var CanClose: Boolean);
<------>    procedure
UpdateButtonClick(Sender:
TObject);
<------>    procedure
ConnectionIndClick(Sender:
TObject);
<------>    procedure
RefreshButtonClick(Sender:
TObject);
  private
    DataFileName: string;
  public
<------>    procedure LoadData;
<------>    procedure SaveData;
<------>    procedure UpdateData;
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

const
  BaseFileName =
'EMPLOYEE.ADTG';

procedure TForm1.LoadData;
begin
  DataFileName :=
ExtractFilePath(Paramstr(0))+Base
FileName;
  { If a persisted datafile exists,
assume we exited in a disconnected
    (offline) state and load the data
from the file. }
  if FileExists(DataFileName) then

Employees.LoadFromFile(DataFileN
ame)
  else
  begin
    { Otherwise establish the
connection and get data from the
database }
    ConnectionInd.Checked := True;
    Employees.Open;
```

```
Form1);
  Application.Run;
end.

\\\\\\\\\\\\\\\\
unit BriefcaseMain;

{ This program demonstrates how to
do disconnected briefcase
applications
  with ADO.  When the Connected
checkbox is unchecked the
application is
  switched into offline mode.  If the
application is exited at that point
  then the data is persisted to a file
on disk (along with any edits to the
  data).  When the application is
restarted it will load the persisted
data
  if present, otherwise it will fetch the
data from the database. }

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls, DB, ADODB, Grids,
DBGrids, ExtCtrls;

type
  TForm1 = class(TForm)
    Employees: TADODataSet;
    EmpSource: TDataSource;
    DBGrid1: TDBGrid;
    Connection: TADOConnection;
    Panel1: TPanel;
    ConnectionInd: TCheckBox;
    UpdateButton: TButton;
    RefreshButton: TButton;
    SaveButton: TButton;
<------>    procedure
Form1Create(Sender: TObject);
<------>    procedure
SaveButtonClick(Sender: TObject);
<------>    procedure
Form1CloseQuery(Sender: TObject;
```

```
  try
    { When closing, update the
database if connected or save it to
disk if not }
    if Connection.Connected then
      UpdateData else
      SaveData;
  except
    on E: Exception do
    begin

Application.HandleException(Self);
      CanClose := MessageDlg('Data
not saved/updated, exit anyway?',
        mtConfirmation,
mbYesNoCancel, 0) = mrYes;
    end;
  end;
end;

procedure
TForm1.ConnectionIndClick(Sender
: TObject);
begin
  { Toggle the connection's state }
  if ConnectionInd.Checked then
  begin
    Connection.Open;
    Employees.Connection :=
Connection;
  end else
  begin
    { Note here you must clear the
connection property of the dataset
before
      closing the connection.
Otherwise the dataset will close with
the
      connection. }
    Employees.Connection := nil;
    Connection.Close;
  end;
end;

procedure
TForm1.RefreshButtonClick(Sender
: TObject);
begin
  { Close and reopen the dataset to
```

```
  end;
end;

procedure TForm1.UpdateData;
begin
  { Connect to the database and send
the pending updates }
  ConnectionInd.Checked := True;
  Employees.UpdateBatch;
  DeleteFile(DataFileName);
end;

procedure TForm1.SaveData;
begin
  { Persist the data to disk }

Employees.SaveToFile(DataFileNam
e, pfADTG);
end;

procedure
TForm1.Form1Create(Sender:
TObject);
begin
  Connection.ConnectionString :=
'FILE NAME=' + DataLinkDir +
'\DBDEMOS.UDL';
  LoadData;
end;

procedure
TForm1.SaveButtonClick(Sender:
TObject);
begin
  SaveData;
end;

procedure
TForm1.UpdateButtonClick(Sender:
TObject);
begin
  UpdateData;
end;

procedure
TForm1.Form1CloseQuery(Sender:
TObject; var CanClose: Boolean);
begin
  if Employees.Active then
```

هاتف ٠٩٤٤٥٧٥٣٧١

```
object Form1: TForm1
  Left = 217
  Top = 143
  AutoScroll = False
  Caption = 'Briefcase Demo'
  ClientHeight = 332
  ClientWidth = 571
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = False
  Position = poScreenCenter
  OnCloseQuery = Form1CloseQuery
  OnCreate = Form1Create
  PixelsPerInch = 96
  TextHeight = 13
  object DBGrid1: TDBGrid
    Left = 0
    Top = 38
    Width = 571
    Height = 294
    Align = alClient
    DataSource = EmpSource
    TabOrder = 0
    TitleFont.Charset = DEFAULT_CHARSET
    TitleFont.Color = clWindowText
    TitleFont.Height = -11
    TitleFont.Name = 'MS Sans Serif'
    TitleFont.Style = []
  end
  object Panel1: TPanel
    Left = 0
    Top = 0
    Width = 571
    Height = 38
    Align = alTop
    BevelOuter = bvNone
    TabOrder = 1
    object ConnectionInd: TCheckBox
      Left = 58
      Top = 11
      Width = 79
      Height = 17
```

```
refresh the data.  Note that in this demo
   there is no checking for pending updates so they are lost if you click
   the refresh data button before clicking the Update database button.
}
  ConnectionInd.Checked := True;
  Employees.Close;
  Employees.CommandType := cmdTable;
  Employees.CommandText := 'Employee';
  Employees.Open;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

Object$

الملف النصي لبرنامج

(----Briefcase)

إعداد

علاء الدين اللباد

```
TADOConnection
   ConnectionString =
     'FILE NAME=C:\Program Files\Common Files\System\OLE DB\Data Links' +
     '\DBDemos.UDL'
   LoginPrompt = False
   Mode = cmShareDenyNone
   Provider =
     'C:\Program Files\Common Files\System\OLE DB\Data Links\DBDemos.U' +
     'DL'
   Left = 13
   Top = 3
  end
End
```

```
    Caption = 'Connected'
    TabOrder = 0
    OnClick = ConnectionIndClick
  end
  object UpdateButton: TButton
    Left = 153
    Top = 6
    Width = 84
    Height = 25
    Caption = 'Update Server'
    TabOrder = 1
    OnClick = UpdateButtonClick
  end
  object RefreshButton: TButton
    Left = 250
    Top = 6
    Width = 87
    Height = 25
    Caption = 'Refresh Data'
    TabOrder = 2
    OnClick = RefreshButtonClick
  end
  object SaveButton: TButton
    Left = 352
    Top = 6
    Width = 75
    Height = 25
    Caption = 'Save to disk'
    TabOrder = 3
    OnClick = SaveButtonClick
  end
end
object Employees: TADODataSet
  Connection = Connection
  CursorType = ctStatic
  LockType = ltBatchOptimistic
  CommandText = 'employee'
  CommandType = cmdTable
  Parameters = <>
  Left = 12
  Top = 89
end
object EmpSource: TDataSource
  DataSet = Employees
  Left = 69
  Top = 97
end
object Connection:
```

علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
program Filter;

uses
  Forms,
  DM in 'DM.pas' {DM1},
  CustView in 'CustView.pas' {fmCustView},
  About in 'About.pas' {fmAboutBox},
  Filter1 in 'Filter1.pas' {fmFilterFrm};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TfmCustView, fmCustView);
  Application.CreateForm(TDM1, DM1);
  Application.CreateForm(TfmFilterFrm, fmFilterFrm);
  Application.Run;
end.

\\\\
Unit DM;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  DB, DBTables;

type
  TDM1 = class(TDataModule)
    Customer: TTable;
    CustomerSource: TDataSource;
    SQLCustomer: TQuery;
    SQLOrders: TQuery;
    OrdersSource: TDataSource;
```

Procedures 39

الملف التنفيذي لبرنامج

(Filter)

إعداد

```
TDM1.SQLOrdersFilterRecord(DataSet: TDataSet;
  var Accept: Boolean);
begin
  { This is only called if the Filtered property is True, set
    dynamically by the CheckBox on the CustView form. }
  Accept := SQLOrdersAmountPaid.Value >= OrdersFilterAmount;
end;

end.

\\\\

unit About;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls;

type
  TfmAboutBox = class(TForm)
    Panel1: TPanel;
    ProgramIcon: TImage;
    ProductName: TLabel;
    Version: TLabel;
    Memo1: TMemo;
    OKButton: TButton;
  end;

var
  fmAboutBox: TfmAboutBox;

implementation

{$R *.dfm}

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
unit About;
```

```
    SQLOrdersOrderNo: TFloatField;
    SQLOrdersCustno: TFloatField;
    SQLOrdersSaleDate: TDateTimeField;
    SQLOrdersShipDate: TDateTimeField;
    SQLOrdersEmpNo: TIntegerField;
    SQLOrdersAmountPaid: TCurrencyField;
    <------>   procedure DM1Create(Sender: TObject);
    <------>   procedure SQLOrdersFilterRecord(DataSet: TDataSet;
      var Accept: Boolean);
  public
    { The variable below will be accessible to to CustView (because it is
      public and this unit is in its uses). It is used in
      SQLOrdersFilterRecord to set the Filter amount for the Orders Query. }
    OrdersFilterAmount: Extended;
  end;

var
  DM1: TDM1;

implementation

{$R *.dfm}

<------>   procedure TDM1.DM1Create(Sender: TObject);
begin
  try
    Screen.Cursor := crHourGlass;
    SQLCustomer.Open;
  finally
    Screen.Cursor := crDefault;
  end;
end;

<------>   procedure
```

```
  About1: TMenuItem;
  DBNavigator1: TDBNavigator;
  Label2: TLabel;
  rgDataSet: TRadioGroup;
  SpeedButton1: TSpeedButton;
  DBGrid1: TDBGrid;
  GroupBox1: TGroupBox;
  cbFilterOrders: TCheckBox;
  Label1: TLabel;
  Edit1: TEdit;
  <------> procedure
rgDataSetClick(Sender: TObject);
  <------> procedure
SpeedButton1Click(Sender:
TObject);
  <------> procedure
cbFilterOrdersClick(Sender:
TObject);
  <------> procedure
About1Click(Sender: TObject);
  <------> procedure
Edit1Change(Sender: TObject);
  <------> procedure
DBGrid1Enter(Sender: TObject);
  <------> procedure
DBGrid2Enter(Sender: TObject);
 end;

var
 fmCustView: TfmCustView;

implementation

uses DM, Filter1, About;

{$R *.dfm}

{ Change the Dataset for the
Customer from Query to Table,
 or Table to Query. If a filter is
current, set that filter
 to the just-changed Dataset. }
<------> procedure
TfmCustView.rgDataSetClick(Sende
r: TObject);
var
 st: string;
begin
 with DM1, CustomerSource do
```

```

interface

uses
 Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
 StdCtrls, ExtCtrls;

type
 TfmAboutBox = class(TForm)
  Panel1: TPanel;
  ProgramIcon: TImage;
  ProductName: TLabel;
  Version: TLabel;
  Memo1: TMemo;
  OKButton: TButton;
 end;

var
 fmAboutBox: TfmAboutBox;

implementation

{$R *.dfm}

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\

unit CustView;

interface

uses
 Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
 ExtCtrls, DBCtrls, Grids, DBGrids,
ComCtrls, StdCtrls, Mask, Buttons,
 Menus;

type
 TfmCustView = class(TForm)
  StatusBar1: TStatusBar;
  DBGrid2: TDBGrid;
  Label3: TLabel;
  MainMenu1: TMainMenu;
```

```
  Edit1Change(nil); { To make sure the value is set. }
end;

<------>  procedure TfmCustView.About1Click(Sender: TObject);
begin
 with TFMAboutBox.Create(nil) do
 try
  ShowModal;
 finally
  Free;
 end;
end;

<------>  procedure TfmCustView.Edit1Change(Sender: TObject);
begin
 { We must test of Edit1.Text not being blank, or an exception will
  be raised when the user clears the contents of the Edit. }
 if (cbFilterOrders.checked) and (Edit1.Text <> '') then
  try
   { Attempt to set the variable on DM1 that holds the filter amount. }
   DM1.OrdersFilterAmount := StrToFloat(fmCustView.Edit1.Text);

   { Refreshing SQLOrders will cause DM1.SQLOrdersFilterRecord to be called
    once for each orders record for the current Customer. }
   DM1.SQLOrders.Refresh;

  except
   on EConvertError do
    { This happens before DM1.SQLOrders.refresh if the user typed in a
     non-numeric value, thus preventing it from trying to use an invalid
     value to filter itself, which
```

```
 begin
  { Is the other Dataset Filtered? Get its filter. }
  if Dataset.Filtered then
   st := Dataset.Filter;
  case rgDataset.ItemIndex of
   0: if Dataset <> SQLCustomer then
      Dataset := SQLCustomer;
   1: if CustomerSource.Dataset <> Customer then
      Dataset := Customer;
  end;

  { Set the Filter of the current Dataset. }
  if st <> '' then
  begin
   Dataset.Filter := st;
   Dataset.Filtered := True;
  end;
 end;
end;

<------>  procedure TfmCustView.SpeedButton1Click(Sender: TObject);
begin
 fmFilterFrm.Show;
end;

<------>  procedure TfmCustView.cbFilterOrdersClick(Sender: TObject);
begin
 { DM1.SQLOrders.Filtered will be true if the box is checked, false otherwise. }
 DM1.SQLOrders.Filtered := cbFilterOrders.Checked;

 { If the number changed while the box is unchecked, DM1.OrdersFilterAmount
  won't know about it.  The Edit1Change <------>  procedure will set the value and
  apply the filter.}
 if cbFilterOrders.Checked then
```

```
   Label3: TLabel;
   Memo1: TMemo;
   GroupBox1: TGroupBox;
   cbCaseSensitive: TCheckBox;
   cbNoPartialCompare: TCheckBox;
   ComboBox1: TComboBox;
   Label4: TLabel;
   BtnApplyFilter: TButton;
   BtnClear: TButton;
   BtnClose: TButton;
   <------>  procedure AddFieldName(Sender: TObject);
   <------>  procedure ListBox2DblClick(Sender: TObject);
   <------>  procedure ApplyFilter(Sender: TObject);
   <------>  procedure FormCreate(Sender: TObject);
   <------>  procedure Memo1Change(Sender: TObject);
   <------>  procedure cbCaseSensitiveClick(Sender: TObject);
   <------>  procedure cbNoPartialCompareClick(Sender: TObject);
   <------>  procedure SBtnClearClick(Sender: TObject);
   <------>  procedure ComboBox1Change(Sender: TObject);
   <------>  procedure SBtnCloseClick(Sender: TObject);
  end;

var
  fmFilterFrm: TfmFilterFrm;

implementation

uses DM, CustView;

{$R *.dfm}

{ Adds current listbox field name to memo. }
<------>  procedure TfmFilterFrm.AddFieldName(Sende
```

```
would raise an exception once for
        each record displayed.}
      raise Exception.Create('Threshold Amount must be a number')
    end
end;

{ Set the navigator to the Customer Datasource. }
<------>  procedure TfmCustView.DBGrid1Enter(Sender: TObject);
begin
  DBNavigator1.DataSource := DBGrid1.DataSource;
end;

{ Set the navigator to the Orders Datasource. }
<------>  procedure TfmCustView.DBGrid2Enter(Sender: TObject);
begin
  DBNavigator1.DataSource := DBGrid2.DataSource;
end;

end.


\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

unit Filter1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Buttons, DB;

type
  TfmFilterFrm = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    ListBox1: TListBox;
    ListBox2: TListBox;
```

```
TObject);
var
  I: Integer;
begin
  for I := 0 to
DM1.CustomerSource.Dataset.Field
Count - 1 do

ListBox1.Items.Add(DM1.Customer.
Fields[I].FieldName);

  { Add date dependent Query
ComboBox1. }

ComboBox1.Items.Add('LastInvoice
Date >= ''' +
    DateToStr(EncodeDate(1994, 09,
30)) + '''');
  ComboBox1.Items.Add('Country =
''US'' and LastInvoiceDate > ''' +
    DateToStr(EncodeDate(1994, 06,
30)) + '''');
end;

{ Since the Filter property is a
TStrings and the Memo field
  is a TMemo, convert the Memo's
wrapped text to a string,
  which is then used when the user
presses Apply. }
<------>   procedure
TfmFilterFrm.Memo1Change(Sende
r: TObject);
var
 I: Integer;
begin
  ComboBox1.Text :=
Memo1.Lines[0];
  for I := 1 to Memo1.Lines.Count - 1
do
    ComboBox1.Text :=
ComboBox1.Text + ' ' +
Memo1.Lines[I];
end;

{ Set the Customer's Dataset Case
Sensitive Filter Option. }
<------>   procedure
TfmFilterFrm.cbCaseSensitiveClick(
```

```
r: TObject);
begin
  if Memo1.Text <> '' then
    Memo1.Text := Memo1.Text + ' ';
  Memo1.Text := Memo1.Text +
ListBox1.Items[ListBox1.ItemIndex]
;
end;

{ Adds current Filter operator to
memo. }
<------>   procedure
TfmFilterFrm.ListBox2DblClick(Se
nder: TObject);
begin
  if Memo1.Text <> '' then
    Memo1.Text := Memo1.Text + ' '+
ListBox2.Items[ListBox2.ItemIndex]
;
end;

<------>   procedure
TfmFilterFrm.ApplyFilter(Sender:
TObject);
begin
  with DM1.CustomerSource.Dataset
do
  begin
    if ComboBox1.Text <> '' then
    begin
      Filter := ComboBox1.Text;
      Filtered := True;
      fmCustView.Caption :=
'Customers - Filtered';
    end
    else begin
      Filter := '';
      Filtered := False;
      fmCustView.Caption :=
'Customers - Unfiltered'
    end;
  end;
end;

{ Populate the ListBox1 with
available fields from the Customer
Dataset. }
<------>   procedure
TfmFilterFrm.FormCreate(Sender:
```

```
ender: TObject);
begin
  Memo1.Lines.Clear;
  
Memo1.Lines.Add(ComboBox1.Text
);
end;

{ Close the Filter Form. }
<------>  procedure
TfmFilterFrm.SBtnCloseClick(Send
er: TObject);
begin
  Close;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

```
<------>  ObjectS50
```

الملف النصي لبرنامج

(Filter)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<------>  Object fmCustView:
TfmCustView
  Left = 227
  Top = 109
  BorderStyle = bsSingle
  Caption = 'Customers - Unfiltered'
  ClientHeight = 362
  ClientWidth = 493
  Color = clBtnFace
  ParentFont = True
  Menu = MainMenu1
  OldCreateOrder = True
  PixelsPerInch = 96
  TextHeight = 13
  <------>  Object Label3: TLabel
```

```
Sender: TObject);
begin
  with DM1.CustomerSource.Dataset
do
    if cbCaseSensitive.checked then
      FilterOptions := FilterOptions -
[foCaseInSensitive]
    else
      FilterOptions := FilterOptions +
[foCaseInsensitive];
end;

{ Set the Customer Partial Compare
Filter Option. }
<------>  procedure
TfmFilterFrm.cbNoPartialCompare
Click(Sender: TObject);
begin
  with DM1.CustomerSource.Dataset
do
    if cbNoPartialCompare.checked
then
      FilterOptions := FilterOptions +
[foNoPartialCompare]
    else
      FilterOptions := FilterOptions -
[foNoPartialCompare];
end;

{ Add User-Entered filters into list
box at runtime. }
<------>  procedure
TfmFilterFrm.SBtnClearClick(Send
er: TObject);
var
  st: string;
begin
  Memo1.Lines.Clear;
  st := ComboBox1.Text;
  ComboBox1.Text := '';
  if ComboBox1.Items.IndexOf(st) =
-1 then
    ComboBox1.Items.Add(st);
end;

{ Reset the Memo field when the
Filter ComboBox changes. }
<------>  procedure
TfmFilterFrm.ComboBox1Change(S
```

Top = 208
Width = 350
Height = 129
DataSource = DM1.OrdersSource
TabOrder = 1
TitleFont.Charset = DEFAULT_CHARSET
TitleFont.Color = clWindowText
TitleFont.Height = -11
TitleFont.Name = 'MS Sans Serif'
TitleFont.Style = []
OnEnter = DBGrid2Enter
Columns = <
item
Expanded = False
FieldName = 'OrderNo'
Visible = True
end
item
Expanded = False
FieldName = 'Custno'
Title.Caption = 'CustNo'
Width = 45
Visible = True
end
item
Expanded = False
FieldName = 'AmountPaid'
Title.Caption = 'Amount Paid'
Width = 122
Visible = True
end
item
Expanded = False
FieldName = 'ShipDate'
Title.Caption = 'Ship Date'
Width = 83
Visible = True
end>
end
<------> Object DBNavigator1: TDBNavigator
Left = 132
Top = 4
Width = 88
Height = 25
DataSource = DM1.CustomerSource

Left = 12
Top = 192
Width = 93
Height = 16
Caption = 'Orders Query'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
end
<------> Object Label2: TLabel
Left = 8
Top = 36
Width = 74
Height = 16
Caption = 'Customers'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
end
<------> Object SpeedButton1: TSpeedButton
Left = 9
Top = 128
Width = 96
Height = 25
Caption = 'Filter &Customers'
NumGlyphs = 2
OnClick = SpeedButton1Click
end
<------> Object StatusBar1: TStatusBar
Left = 0
Top = 343
Width = 493
Height = 19
Panels = <>
end
<------> Object DBGrid2: TDBGrid
Left = 132

```
      Expanded = False
      FieldName = 'Country'
      Width = 64
      Visible = True
    end
    item
      Expanded = False
      FieldName = 'LastInvoiceDate'
      Title.Caption = 'Last Invoice'
      Width = 64
      Visible = True
    end
    item
      Expanded = False
      FieldName = 'Contact'
      Width = 64
      Visible = True
    end
    item
      Expanded = False
      FieldName = 'City'
      Width = 64
      Visible = True
    end
    item
      Expanded = False
      FieldName = 'State'
      Visible = True
    end>
  end
  <------>  Object GroupBox1:
TGroupBox
   Left = 7
   Top = 224
   Width = 121
   Height = 81
   Caption = 'Orders Filter'
   TabOrder = 5
   <------>  Object Label1: TLabel
    Left = 4
    Top = 52
    Width = 60
    Height = 13
    Caption = 'Amount &Paid'
    FocusControl = Edit1
   end
   <------>  Object cbFilterOrders:
TCheckBox
```

```
  VisibleButtons = [nbFirst,
nbPrior, nbNext, nbLast]
  TabOrder = 3
 end
 <------>  Object rgDataSet:
TRadioGroup
  Left = 8
  Top = 60
  Width = 97
  Height = 61
  Caption = 'Dataset'
  ItemIndex = 1
  Items.Strings = (
   '&Query-Based'
   '&Table-Based')
  TabOrder = 2
  OnClick = rgDataSetClick
 end
 <------>  Object DBGrid1:
TDBGrid
  Left = 132
  Top = 33
  Width = 350
  Height = 163
  DataSource =
DM1.CustomerSource
  TabOrder = 0
  TitleFont.Charset =
DEFAULT_CHARSET
  TitleFont.Color = clWindowText
  TitleFont.Height = -11
  TitleFont.Name = 'MS Sans Serif'
  TitleFont.Style = []
  OnEnter = DBGrid1Enter
  Columns = <
   item
    Expanded = False
    FieldName = 'CustNo'
    Width = 43
    Visible = True
   end
   item
    Expanded = False
    FieldName = 'Company'
    Width = 64
    Visible = True
   end
   item
```

```
 <------> Object CustomerSource: TDataSource
   DataSet = Customer
   Left = 111
   Top = 33
 end
 <------> Object SQLCustomer: TQuery
   DatabaseName = 'DBDEMOS'
   SQL.Strings = (
     'SELECT * FROM "CUSTOMER.DB" ')
   Left = 26
   Top = 62
 end
 <------> Object SQLOrders: TQuery
   Active = True
   DatabaseName = 'DBDEMOS'
   OnFilterRecord = SQLOrdersFilterRecord
   DataSource = CustomerSource
   SQL.Strings = (
     'Select * From Orders Where Custno = :CustNo ')
   Left = 196
   Top = 32
   ParamData = <
     item
       DataType = ftFloat
       Name = 'CustNo'
       ParamType = ptUnknown
       Size = 8
     end>
   <------> Object SQLOrdersOrderNo: TFloatField
     DisplayWidth = 10
     FieldName = 'OrderNo'
   end
   <------> Object SQLOrdersCustno: TFloatField
     DisplayWidth = 8
     FieldName = 'Custno'
   end
   <------> Object SQLOrdersAmountPaid: TCurrencyField
     DisplayWidth = 13
```

```
     Left = 8
     Top = 20
     Width = 109
     Height = 17
     Caption = '&Filter'
     TabOrder = 0
     OnClick = cbFilterOrdersClick
   end
   <------> Object Edit1: TEdit
     Left = 68
     Top = 48
     Width = 49
     Height = 21
     TabOrder = 1
     Text = '1000'
     OnChange = Edit1Change
   end
 end
 <------> Object MainMenu1: TMainMenu
   Left = 104
   Top = 5
   <------> Object About1: TMenuItem
     Caption = '&About'
     OnClick = About1Click
   end
 end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

<------> Object DM1: TDM1
 OldCreateOrder = True
 OnCreate = DM1Create
 Left = 175
 Top = 163
 Height = 223
 Width = 451
 <------> Object Customer: TTable
   Active = True
   DatabaseName = 'DBDEMOS'
   Filtered = True
   IndexFieldNames = 'Company'
   TableName = 'CUSTOMER.DB'
   Left = 27
   Top = 9
 end
```

Left = 8
Top = 128
Width = 27
Height = 13
Caption = '&Fields'
FocusControl = ListBox1
end
<------> Object Label2: TLabel
Left = 8
Top = 44
Width = 44
Height = 13
Caption = '&Condition'
end
<------> Object Label3: TLabel
Left = 161
Top = 128
Width = 46
Height = 13
Caption = '&Operators'
end
<------> Object Label4: TLabel
Left = 8
Top = 8
Width = 16
Height = 13
Caption = '&List'
FocusControl = ComboBox1
end
<------> Object ListBox1: TListBox
Left = 8
Top = 144
Width = 143
Height = 73
Hint = 'Doubleclick to add'
TabStop = False
ItemHeight = 13
ParentShowHint = False
ShowHint = True
Sorted = True
TabOrder = 2
OnDblClick = AddFieldName
end
<------> Object ListBox2: TListBox
Left = 161
Top = 144

FieldName = 'AmountPaid'
end
<------> Object SQLOrdersSaleDate: TDateTimeField
Alignment = taRightJustify
DisplayWidth = 10
FieldName = 'SaleDate'
end
<------> Object SQLOrdersShipDate: TDateTimeField
Alignment = taRightJustify
DisplayWidth = 10
FieldName = 'ShipDate'
end
<------> Object SQLOrdersEmpNo: TIntegerField
DisplayWidth = 8
FieldName = 'EmpNo'
end
end
<------> Object OrdersSource: TDataSource
DataSet = SQLOrders
Left = 272
Top = 32
end
End
\\
<------> Object fmFilterFrm: TfmFilterFrm
Left = 332
Top = 132
ActiveControl = Memo1
BorderStyle = bsSingle
Caption = 'Filter Condition'
ClientHeight = 298
ClientWidth = 305
Color = clBtnFace
ParentFont = True
OldCreateOrder = True
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<------> Object Label1: TLabel

```
cbNoPartialCompare: TCheckBox
    Left = 8
    Top = 36
    Width = 125
    Height = 17
    Hint = 'Disable partial string
comparisons (strings ending with
''*'')'
    Caption = 'No Partial Compare'
    TabOrder = 1
    OnClick =
cbNoPartialCompareClick
  end
 end
 <------>  Object ComboBox1:
TComboBox
  Left = 8
  Top = 24
  Width = 289
  Height = 21
  Cursor = crArrow
  ItemHeight = 13
  TabOrder = 0
  OnChange = ComboBox1Change
  Items.Strings = (
    'Company = ''S*'''
    'Country <> ''US'''
    'Country = ''US''')
 end
 <------>  Object BtnApplyFilter:
TButton
  Left = 224
  Top = 143
  Width = 75
  Height = 25
  Caption = '&Apply'
  TabOrder = 5
  OnClick = ApplyFilter
 end
 <------>  Object BtnClear: TButton
  Left = 224
  Top = 173
  Width = 75
  Height = 25
  Caption = 'Cl&ear'
  TabOrder = 6
  OnClick = SBtnClearClick
 end
```

```
   Width = 53
   Height = 141
   Hint = 'Double-click to add'
   TabStop = False
   ItemHeight = 13
   Items.Strings = (
     '>'
     '<'
     '= '
     '>='
     '<='
     '<>'
     'AND'
     'OR'
     '('
     ')')
   ParentShowHint = False
   ShowHint = True
   TabOrder = 3
   OnDblClick = ListBox2DblClick
 end
 <------>  Object Memo1: TMemo
   Left = 8
   Top = 60
   Width = 289
   Height = 65
   TabOrder = 1
   OnChange = Memo1Change
 end
 <------>  Object GroupBox1:
TGroupBox
   Left = 12
   Top = 224
   Width = 139
   Height = 61
   Caption = 'F&ilter Options'
   TabOrder = 4
   <------>  Object cbCaseSensitive:
TCheckBox
     Left = 8
     Top = 16
     Width = 97
     Height = 17
     Caption = 'Case Sensitive'
     TabOrder = 0
     OnClick = cbCaseSensitiveClick
   end
   <------>  Object
```

<------> Object ProductName: TLabel
Left = 88
Top = 16
Width = 65
Height = 13
Caption = 'Filter Example'
IsControl = True
end
<------> Object Version: TLabel
Left = 88
Top = 36
Width = 44
Height = 13
Caption = 'Version 1'
IsControl = True
end
<------> Object Memo1: TMemo
Left = 8
Top = 72
Width = 265
Height = 181
Lines.Strings = (
'This example illustrates'
''
'- The use of a Data Module with a '
' Query linked through its Datasource'
' property for displaying detail records.'
'- Switching Between Query- and Table-'
' Based datasets.'
'- Filtering Records dynamically, by setting '
' the Filter property of a Dataset.'
'- Filtering Records more statically, '
' by providing a value to a variable hooked'
' to the OnFilterRecord event .'
''
'At runtime, the Data Module''s only master '
'datasource can be connected to either the '

<------> Object BtnClose: TButton
Left = 224
Top = 256
Width = 75
Height = 25
Caption = 'Cl&ose'
TabOrder = 7
OnClick = SBtnCloseClick
end
End
\\
<------> Object fmAboutBox: TfmAboutBox
Left = 442
Top = 133
BorderStyle = bsDialog
Caption = 'fmAboutBox'
ClientHeight = 321
ClientWidth = 299
Color = clBtnFace
ParentFont = True
OldCreateOrder = True
Position = poScreenCenter
PixelsPerInch = 96
TextHeight = 13
<------> Object Panel1: TPanel
Left = 8
Top = 8
Width = 281
Height = 265
BevelInner = bvRaised
BevelOuter = bvLowered
Caption = 'Panel1'
TabOrder = 0
<------> Object ProgramIcon: TImage
Left = 8
Top = 8
Width = 65
Height = 57
Picture.Data = {
0000}
Stretch = True
IsControl = True
end

الملفالتنفيذي لبرنامج

(GdsDemo)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
program GdsDemo;

uses
  Forms,
  GdsStd in 'GdsStd.pas' {GDSStdForm},
  GdsData in 'GdsData.pas' {StdDataForm},
  RecForm in 'RecForm.pas' {RecViewForm},
  GridForm in 'GridForm.pas' {GridViewForm};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TGridViewForm, GridViewForm);
  Application.CreateForm(TRecViewForm, RecViewForm);
  Application.Run;
end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

unit GdsStd;

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls;

type
  TGDSStdForm = class(TForm)
```

```
      'Customer TQuery or the Customer TTable. '
      'This Dataset is linked to a single Datasource '
      'which serves as the link to another dataset,'
      'defined by a dynamic query of the Orders table.'
      ''
      'Filtering is done on the Customer records '
      'dynamically, with the user providing an'
      'arbitrary expression. In contrast, the Filter '
      'button for the Orders table is done in a '
      'more static, code-based fashion, with the'
      'user simply providing a threshold for the '
      'Total Order Amount. An exception is raised'
      'when user entries are inappropriate.')
    ReadOnly = True
    ScrollBars = ssVertical
    TabOrder = 0
   end
  end
  <------>  Object OKButton: TButton
   Left = 111
   Top = 284
   Width = 77
   Height = 27
   Caption = 'OK'
   Default = True
   ModalResult = 1
   TabOrder = 1
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

24 proc

```
var
  GridViewForm: TGridViewForm;

implementation

uses RecForm;

{$R *.dfm}

<------>   procedure TGridViewForm.DBGrid1DblClick(Sender: TObject);
begin
  inherited;
  RecViewForm.Show;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
unit RecForm;

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls, Forms, Dialogs,
  DBCtrls, DB, DBTables, ExtCtrls, StdCtrls, Buttons, Mask, Menus,
  ComCtrls, GdsData;

type
  TRecViewForm = class(TStdDataForm)
    GroupBox2: TPanel;
    OrderNo: TDBEdit;
    CustName: TDBEdit;
    SaleDate: TDBEdit;
    AmountDue: TDBEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label7: TLabel;
    DBEdit3: TDBEdit;
    Label8: TLabel;
    GDSStdPanel: TPanel;
    GDSStdImage: TImage;
    GDSLabel: TLabel;
    GDSLabel2: TLabel;
    GDSSloganLabel: TLabel;
    GDSSloganLabel2: TLabel;
    ImageBevel: TBevel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  GDSStdForm: TGDSStdForm;

implementation

{$R *.dfm}

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

unit GridForm;

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls, Forms, Dialogs,
  Grids, DBGrids, DB, DBTables, ExtCtrls, StdCtrls, Buttons, GdsData;

type
  TGridViewForm = class(TStdDataForm)
    DBGrid1: TDBGrid;
    <------>   procedure DBGrid1DblClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
```

```
    Orders: TTable;
    Cust: TTable;
    OrdersSource: TDataSource;
    OrdersOrderNo: TFloatField;
    OrdersCustNo: TFloatField;
    OrdersSaleDate: TDateTimeField;
    OrdersShipDate:
TDateTimeField;
    OrdersEmpNo: TIntegerField;
    OrdersShipToContact:
TStringField;
    OrdersShipToAddr1:
TStringField;
    OrdersShipToAddr2:
TStringField;
    OrdersShipToCity: TStringField;
    OrdersShipToState: TStringField;
    OrdersShipToZip: TStringField;
    OrdersShipToCountry:
TStringField;
    OrdersShipToPhone:
TStringField;
    OrdersShipVIA: TStringField;
    OrdersPO: TStringField;
    OrdersTerms: TStringField;
    OrdersPaymentMethod:
TStringField;
    OrdersItemsTotal:
TCurrencyField;
    OrdersTaxRate: TFloatField;
    OrdersFreight: TCurrencyField;
    OrdersCustName: TStringField;
    OrdersAmountDue:
TCurrencyField;
    OrdersAmountPaid:
TCurrencyField;
    GroupBox1: TGroupBox;
    FilterOnLabel: TLabel;
    FilterCriteria: TEdit;
    FilterCheckBox: TCheckBox;
    NextBtn: TButton;
    PriorBtn: TButton;
    OrdersTaxAmount:
TCurrencyField;
    <------>  procedure
FilterOnRadioGroupClick(Sender:
TObject);
    <------>  procedure
FormCreate(Sender: TObject);
    DBEdit4: TDBEdit;
    Label5: TLabel;
    DBEdit1: TDBEdit;
    Label9: TLabel;
    DBEdit5: TDBEdit;
    Bevel1: TBevel;
    DBNavigator1: TDBNavigator;
    Panel1: TPanel;
    OnCredit: TDBCheckBox;
    Label11: TLabel;
    DBEdit7: TDBEdit;
    DBEdit8: TDBEdit;
    Label12: TLabel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  RecViewForm: TRecViewForm;

implementation

{$R *.dfm}

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

unit GdsData;

interface

uses
  SysUtils, Windows, Messages,
Classes, Graphics, Controls, Forms,
Dialogs,
  Buttons, StdCtrls, ExtCtrls, Grids,
DBGrids, DB, DBTables, Mask,
  DBCtrls, GdsStd;

type
  TStdDataForm =
class(TGDSStdForm)
    StdCtrlPanel: TPanel;
    FilterOnRadioGroup:
TRadioGroup;
```

```
        Orders.OnFilterRecord :=
OrdersFilterOnDate;
        FilterCriteria.Text :=
DateToStr(FLastDate);
      end;
    1: begin
        Orders.OnFilterRecord :=
OrdersFilterOnAmount;
        FilterCriteria.Text :=
FloatToStr(FLastAmount);
      end;
   end;
   ActiveControl := FilterCriteria;
 end;
 if Orders.Filtered then
Orders.Refresh;
end;

<------>   procedure
TStdDataForm.FormCreate(Sender:
TObject);
begin
 inherited;
 FLastDate := EncodeDate(1995, 1,
1);
 FLastAmount := 1000;
 FilterOnRadioGroup.ItemIndex :=
0;
end;

{ Calculate the value of AmountDue.
Used in the OnCalcFields
 and OnFilterRecord event
handlers. }

function
TStdDataForm.CalcAmountDue:
Double;
begin
 Result :=
   OrdersItemsTotal.Value * (1.0 +
OrdersTaxRate.Value / 100) +
   OrdersFreight.Value -
OrdersAmountPaid.Value;
end;

{ Convert the FilterCriteria text into
a Date or Float.  This value
 will be used in the OnFilterRecord
```

```
   <------>   procedure
OrdersFilterOnDate(DataSet:
TDataSet; var Accept: Boolean);
   <------>   procedure
OrdersFilterOnAmount(DataSet:
TDataSet; var Accept: Boolean);
   <------>   procedure
OrdersCalcFields(DataSet:
TDataSet);
   <------>   procedure
FilterCheckBoxClick(Sender:
TObject);
   <------>   procedure
PriorBtnClick(Sender: TObject);
   <------>   procedure
NextBtnClick(Sender: TObject);
   <------>   procedure
FilterCriteriaExit(Sender: TObject);
   <------>   procedure
FilterCriteriaKeyPress(Sender:
TObject; var Key: Char);
 protected
   FLastAmount: Double;
   FLastDate: TDateTime;
   function CalcAmountDue:
Double;
   <------>   procedure
ConvertFilterCriteria;
 end;

var
 StdDataForm: TStdDataForm;

implementation

{$R *.dfm}

<------>   procedure
TStdDataForm.FilterOnRadioGrou
pClick(Sender: TObject);
begin
 inherited;
 with FilterOnRadioGroup do
 begin
   FilterOnLabel.Caption :=
Format('Records where %S >=',
[Items[ItemIndex]]);
   case ItemIndex of
     0: begin
```

```
fly. Include in the dataset
  if its amount is greater than or equal to the specified filter
  criteria. Note that calculated and lookup fields are undefined
  in an OnFilterRecord event handler. }
<------>   procedure TStdDataForm.OrdersFilterOnAmount(DataSet: TDataSet;
  var Accept: Boolean);
begin
  inherited;
  Accept := CalcAmountDue >= FLastAmount;
end;

{ Include this order in the dataset if its date is greater
  than or equal to the specified filter criteria. }
<------>   procedure TStdDataForm.OrdersFilterOnDate(DataSet: TDataSet;
  var Accept: Boolean);
begin
  inherited;
  Accept := OrdersSaleDate.Value >= FLastDate;
end;

<------>   procedure TStdDataForm.OrdersCalcFields(DataSet: TDataSet);
begin
  inherited;
  OrdersTaxAmount.Value := OrdersItemsTotal.Value * (OrdersTaxRate.Value / 100);
  OrdersAmountDue.Value := CalcAmountDue;
end;

{ Store contents of filter criteria from edit control }
<------>   procedure TStdDataForm.FilterCheckBoxClick(Sender: TObject);
begin
```

```
callback instead of using the
  FilterCriteria directly, so that the string does not need to be
  converted each time the event is triggered. }

<------>   procedure TStdDataForm.ConvertFilterCriteria;
begin
  if FilterCriteria.Text <> '' then
    case FilterOnRadioGroup.ItemIndex of
      0: FLastDate := StrToDate(FilterCriteria.Text);
      1: FLastAmount := StrToFloat(FilterCriteria.Text);
    end;
  if Orders.Filtered then Orders.Refresh;
end;

{ Try to convert the filter criteria whenever the edit control
  looses focus, or the user presses enter }

<------>   procedure TStdDataForm.FilterCriteriaExit(Sender: TObject);
begin
  inherited;
  ConvertFilterCriteria;
end;

<------>   procedure TStdDataForm.FilterCriteriaKeyPress(Sender: TObject;
  var Key: Char);
begin
  inherited;
  if Key = #13 then
  begin
    ConvertFilterCriteria;
    Key := #0
  end;
end;

{ Calculate this order's total on the
```

```
ClientHeight = 322
ClientWidth = 480
PixelsPerInch = 96
TextHeight = 13
inherited GDSStdPanel: TPanel
  Width = 480
end
inherited StdCtrlPanel: TPanel
  Width = 480
  inherited PriorBtn: TButton
    Left = 385
  end
end
<------> Object GroupBox2: TPanel [2]
  Left = 0
  Top = 161
  Width = 480
  Height = 161
  Align = alClient
  BevelOuter = bvLowered
  TabOrder = 2
  <------> Object Label1: TLabel
    Left = 8
    Top = 42
    Width = 46
    Height = 13
    Caption = 'Order No:'
  end
  <------> Object Label2: TLabel
    Left = 8
    Top = 68
    Width = 47
    Height = 13
    Caption = 'Customer:'
  end
  <------> Object Label3: TLabel
    Left = 131
    Top = 41
    Width = 50
    Height = 13
    Caption = 'Sale Date:'
  end
  <------> Object Label4: TLabel
    Left = 300
    Top = 126
    Width = 62
    Height = 13
```

```
  inherited;
  ConvertFilterCriteria;
  Orders.Filtered :=
FilterCheckBox.Checked;
  NextBtn.Enabled := not
FilterCheckBox.Checked;
  PriorBtn.Enabled := not
FilterCheckBox.Checked;
end;

{ Button handlers for new filter-oriented dataset navigation methods
}
<------> procedure
TStdDataForm.PriorBtnClick(Sender: TObject);
begin
  inherited;
  ConvertFilterCriteria;
  Orders.FindPrior;
end;

<------> procedure
TStdDataForm.NextBtnClick(Sender: TObject);
begin
  inherited;
  ConvertFilterCriteria;
  Orders.FindNext;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

<------> Objects27

الملف النصي لبرنامج

(GdsDemo)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
inherited RecViewForm: TRecViewForm
  Caption = 'Single Record View'
```

```
    DataSource = OrdersSource
    TabOrder = 0
  end
  <------>  Object CustName:
TDBEdit
    Left = 66
    Top = 65
    Width = 187
    Height = 21
    DataField = 'CustName'
    DataSource = OrdersSource
    TabOrder = 1
  end
  <------>  Object SaleDate:
TDBEdit
    Left = 186
    Top = 39
    Width = 67
    Height = 21
    DataField = 'SaleDate'
    DataSource = OrdersSource
    TabOrder = 2
  end
  <------>  Object AmountDue:
TDBEdit
    Left = 376
    Top = 123
    Width = 81
    Height = 21
    DataField = 'Amount'
    DataSource = OrdersSource
    TabOrder = 3
  end
  <------>  Object DBEdit3:
TDBEdit
    Left = 376
    Top = 15
    Width = 81
    Height = 21
    DataField = 'ItemsTotal'
    DataSource = OrdersSource
    TabOrder = 4
  end
  <------>  Object DBEdit4:
TDBEdit
    Left = 376
    Top = 87
    Width = 81
```

```
    Caption = 'Amount Due:'
  end
  <------>  Object Label7: TLabel
    Left = 300
    Top = 19
    Width = 55
    Height = 13
    Caption = 'Items Total:'
    FocusControl = DBEdit3
  end
  <------>  Object Label8: TLabel
    Left = 300
    Top = 92
    Width = 63
    Height = 13
    Caption = 'Amount Paid:'
    FocusControl = DBEdit4
  end
  <------>  Object Label5: TLabel
    Left = 300
    Top = 67
    Width = 35
    Height = 13
    Caption = 'Freight:'
    FocusControl = DBEdit1
  end
  <------>  Object Label9: TLabel
    Left = 300
    Top = 43
    Width = 21
    Height = 13
    Caption = 'Tax:'
    FocusControl = DBEdit5
  end
  <------>  Object Bevel1: TBevel
    Left = 309
    Top = 115
    Width = 148
    Height = 2
    Shape = bsTopLine
  end
  <------>  Object OrderNo:
TDBEdit
    Left = 66
    Top = 39
    Width = 50
    Height = 21
    DataField = 'OrderNo'
```

```
      Height = 13
      Caption = 'Terms:'
      FocusControl = DBEdit7
    end
    <------>  Object Label12: TLabel
      Left = 23
      Top = 10
      Width = 18
      Height = 13
      Caption = 'PO:'
      FocusControl = DBEdit8
    end
    <------>  Object OnCredit: TDBCheckBox
      Left = 96
      Top = 32
      Width = 77
      Height = 17
      Caption = 'On Credit'
      DataField = 'PaymentMethod'
      DataSource = OrdersSource
      TabOrder = 0
      ValueChecked = 'Credit'
      ValueUnchecked = 'Visa;Cash;Check;MC;Amex'
    end
    <------>  Object DBEdit7: TDBEdit
      Left = 44
      Top = 30
      Width = 40
      Height = 21
      DataField = 'Terms'
      DataSource = OrdersSource
      TabOrder = 1
    end
    <------>  Object DBEdit8: TDBEdit
      Left = 44
      Top = 6
      Width = 113
      Height = 21
      DataField = 'PO'
      DataSource = OrdersSource
      TabOrder = 2
    end
  end
end
```

```
      Height = 21
      DataField = 'AmountPaid'
      DataSource = OrdersSource
      TabOrder = 5
    end
    <------>  Object DBEdit1: TDBEdit
      Left = 376
      Top = 63
      Width = 81
      Height = 21
      DataField = 'Freight'
      DataSource = OrdersSource
      TabOrder = 6
    end
    <------>  Object DBEdit5: TDBEdit
      Left = 376
      Top = 39
      Width = 81
      Height = 21
      DataField = 'TaxAmount'
      DataSource = OrdersSource
      TabOrder = 7
    end
    <------>  Object DBNavigator1: TDBNavigator
      Left = 107
      Top = 10
      Width = 88
      Height = 21
      DataSource = OrdersSource
      VisibleButtons = [nbFirst, nbPrior, nbNext, nbLast]
      TabOrder = 8
    end
    <------>  Object Panel1: TPanel
      Left = 71
      Top = 92
      Width = 177
      Height = 57
      BevelInner = bvRaised
      BevelOuter = bvLowered
      TabOrder = 9
      <------>  Object Label11: TLabel
        Left = 7
        Top = 34
        Width = 32
```

Color = clBtnFace
ParentFont = True
OldCreateOrder = True
Position = poScreenCenter
OnCloseQuery = FormCloseQuery
OnCreate = FormCreate
OnDestroy = FormDestroy
PixelsPerInch = 96
TextHeight = 13
<-------> object Label1: TLabel
Left = 16
Top = 16
Width = 74
Height = 13
Caption = '&Database Alias:'
FocusControl = AliasCombo
end
<-------> object Label2: TLabel
Left = 16
Top = 56
Width = 31
Height = 13
Caption = '&Query:'
FocusControl = QueryEdit
end
<-------> object Label3: TLabel
Left = 152
Top = 16
Width = 56
Height = 13
Caption = '&User Name:'
FocusControl = NameEdit
end
<-------> object Label4: TLabel
Left = 304
Top = 16
Width = 49
Height = 13
Caption = '&Password:'
FocusControl = PasswordEdit
end
<-------> object Label5: TLabel
Left = 15
Top = 259
Width = 72
Height = 13
Caption = '&Retrieve query:'
FocusControl =

End
\\
inherited GridViewForm: TGridViewForm
Caption = 'Grid View'
PixelsPerInch = 96
TextHeight = 13
<------> Object DBGrid1: TDBGrid [2]
Left = 0
Top = 161
Width = 460
Height = 159
Align = alClient
DataSource = OrdersSource
TabOrder = 2
TitleFont.Charset = DEFAULT_CHARSET
TitleFont.Color = clBlack
TitleFont.Height = -11
TitleFont.Name = 'MS Sans Serif'
TitleFont.Style = []
OnDblClick = DBGrid1DblClick
end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

<-------> object s31
الملف النصي لبرنامج
(BkQuery----)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

<-------> object AdhocForm: TAdhocForm
Left = 144
Top = 297
BorderStyle = bsSingle
Caption = 'Adhoc Background Query Demo'
ClientHeight = 297
ClientWidth = 545

```
    TabOrder = 2
    Text = 'masterkey'
  end
  <-------> object ExecuteBtn: TButton
    Left = 464
    Top = 8
    Width = 75
    Height = 25
    Caption = '&Execute'
    Default = True
    TabOrder = 5
    OnClick = ExecuteBtnClick
  end
  <-------> object CloseBtn: TButton
    Left = 464
    Top = 134
    Width = 75
    Height = 25
    Cancel = True
    Caption = '&Close'
    TabOrder = 6
    OnClick = CloseBtnClick
  end
  <-------> object SavedQueryCombo: TComboBox
    Left = 96
    Top = 256
    Width = 345
    Height = 21
    Style = csDropDownList
    ItemHeight = 13
    TabOrder = 4
    OnChange = SavedQueryComboChange
  end
  <-------> object SaveBtn: TButton
    Left = 464
    Top = 72
    Width = 75
    Height = 25
    Caption = '&Save'
    TabOrder = 7
    OnClick = SaveBtnClick
  end
  <-------> object SaveAsBtn: TButton
    Left = 464
```

```
SavedQueryCombo
  end
  <-------> object Bevel1: TBevel
    Left = 8
    Top = 8
    Width = 449
    Height = 281
    Shape = bsFrame
  end
  <-------> object AliasCombo: TComboBox
    Left = 16
    Top = 32
    Width = 129
    Height = 21
    ItemHeight = 13
    TabOrder = 0
    Text = 'AliasCombo'
  end
  <-------> object QueryEdit: TMemo
    Left = 16
    Top = 72
    Width = 425
    Height = 177
    Lines.Strings = (
      'select Company, Sum(ItemsTotal) - Sum(AmountPaid) as AmountDue '
      '  from customer, orders '
      '  where Customer.CustNo = Orders.CustNo'
      '  group by Company')
    TabOrder = 3
  end
  <-------> object NameEdit: TEdit
    Left = 152
    Top = 32
    Width = 145
    Height = 21
    TabOrder = 1
  end
  <-------> object PasswordEdit: TEdit
    Left = 304
    Top = 32
    Width = 137
    Height = 21
    PasswordChar = '*'
```

```
    Align = alBottom
    Caption = 'Opening...'
    WordWrap = True
  end
  <-------> object DBGrid1: TDBGrid
    Left = 0
    Top = 13
    Width = 435
    Height = 272
    Align = alClient
    DataSource = DataSource
    TabOrder = 0
    TitleFont.Charset =
DEFAULT_CHARSET
    TitleFont.Color = clWindowText
    TitleFont.Height = -11
    TitleFont.Name = 'MS Sans Serif'
    TitleFont.Style = []
  end
  <-------> object DataSource:
TDataSource
    Left = 112
    Top = 248
  end
  <-------> object Query: TQuery
    Left = 80
    Top = 248
  end
  <-------> object Session: TSession
    Left = 16
    Top = 248
  end
  <-------> object Database:
TDatabase
    LoginPrompt = False
    SessionName = 'Default'
    Left = 48
    Top = 248
  end
End

\\\\\\\\\\\\\\\\\\\\\\\
3
<-------> object SaveQueryAs:
TSaveQueryAs
  Left = 111
  Top = 149
  BorderStyle = bsDialog
```

```
    Top = 104
    Width = 75
    Height = 25
    Caption = 'Save &as'
    TabOrder = 8
    OnClick = SaveAsBtnClick
  end
  <-------> object NewBtn: TButton
    Left = 464
    Top = 40
    Width = 75
    Height = 25
    Caption = '&New'
    TabOrder = 9
    OnClick = NewBtnClick
  end
End

\\\\\\\\\\\\\\\\\\\
2
<-------> object QueryForm:
TQueryForm
  Left = 170
  Top = 114
  Width = 443
  Height = 332
  Caption = 'Background Query'
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poDefault
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object QueryLabel:
TLabel
    Left = 0
    Top = 0
    Width = 435
    Height = 13
    Align = alTop
    Caption = '[Query text]'
    WordWrap = True
  end
  <-------> object StatusLine: TLabel
    Left = 0
    Top = 285
    Width = 435
    Height = 13
```

```
    ModalResult = 2
    TabOrder = 2
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\
```

الملف التنفيذي لبرنامج

Procedure37

BkQuery)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
{ Demostrates how to execute a query in a background thread.  This
  files contains the main user interface for this program.  The background
  query code is in ResltFrm }

unit QueryFrm;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, IniFiles, ExtCtrls, DBTables;

type
  TAdhocForm = class(TForm)
    AliasCombo: TComboBox;
    Label1: TLabel;
    QueryEdit: TMemo;
    Label2: TLabel;
    NameEdit: TEdit;
    PasswordEdit: TEdit;
    Label3: TLabel;
    Label4: TLabel;
    ExecuteBtn: TButton;
    CloseBtn: TButton;
```

```
Caption = 'Save query as'
ClientHeight = 102
ClientWidth = 343
Color = clBtnFace
ParentFont = True
OldCreateOrder = True
Position = poScreenCenter
PixelsPerInch = 96
TextHeight = 13
<-------> object Bevel1: TBevel
  Left = 8
  Top = 8
  Width = 329
  Height = 57
  Shape = bsFrame
end
<-------> object Label1: TLabel
  Left = 16
  Top = 11
  Width = 60
  Height = 13
  Caption = 'Query name:'
end
<-------> object NameEdit: TEdit
  Left = 16
  Top = 27
  Width = 305
  Height = 21
  TabOrder = 0
  OnChange = NameEditChange
end
<-------> object OKBtn: TButton
  Left = 184
  Top = 72
  Width = 75
  Height = 25
  Caption = 'OK'
  Default = True
  ModalResult = 1
  TabOrder = 1
end
<-------> object CancelBtn: TButton
  Left = 264
  Top = 72
  Width = 75
  Height = 25
  Cancel = True
  Caption = 'Cancel'
```

```
function StrToIniStr(const Str: string): string;
var
  Buffer: array[0..4095] of Char;
  B, S: PChar;
begin
  if Length(Str) > SizeOf(Buffer) then
    raise Exception.Create('String to large to save in INI file');
  S := PChar(Str);
  B := Buffer;
  while S^ <> #0 do
    case S^ of
      #13, #10:
        begin
          if (S^ = #13) and (S[1] = #10) then Inc(S)
          else if (S^ = #10) and (S[1] = #13) then Inc(S);
          B^ := '\';
          Inc(B);
          B^ := 'n';
          Inc(B);
          Inc(S);
        end;
    else
      B^ := S^;
      Inc(B);
      Inc(S);
    end;
  B^ := #0;
  Result := Buffer;
end;

function IniStrToStr(const Str: string): string;
var
  Buffer: array[0..4095] of Char;
  B, S: PChar;
begin
  if Length(Str) > SizeOf(Buffer) then
    raise Exception.Create('String to read from an INI file');
  S := PChar(Str);
  B := Buffer;
```

```
    SavedQueryCombo: TComboBox;
    Label5: TLabel;
    SaveBtn: TButton;
    SaveAsBtn: TButton;
    NewBtn: TButton;
    Bevel1: TBevel;
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure CloseBtnClick(Sender: TObject);
    <-------> procedure ExecuteBtnClick(Sender: TObject);
    <-------> procedure FormDestroy(Sender: TObject);
    <-------> procedure NewBtnClick(Sender: TObject);
    <-------> procedure SaveBtnClick(Sender: TObject);
    <-------> procedure SaveAsBtnClick(Sender: TObject);
    <-------> procedure SavedQueryComboChange(Sender: TObject);
    <-------> procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
  private
    { Private declarations }
    QueryName, OldAlias: string;
    SavedQueries: TIniFile;
    Unnamed: Boolean;
    function IsModified: Boolean;
    function CheckModified: Boolean;
    <-------> procedure Unmodify;
    <-------> procedure ReadQuery;
    <-------> procedure SaveQuery;
    <-------> procedure SaveQueryAs;
  public
    { Public declarations }
  end;

var
  AdhocForm: TAdhocForm;

implementation

uses DB, ResltFrm, SaveQAs;

{$R *.dfm}
```

```
  QueryEdit.Modified := False;
end;

<-------> procedure
TAdhocForm.ReadQuery;
begin
  if not CheckModified then Exit;
  with SavedQueries do
  begin
    QueryName :=
SavedQueryCombo.Items[SavedQue
ryCombo.ItemIndex];
    QueryEdit.Text :=
IniStrToStr(ReadString(QueryNam
e, 'Query', ''));
    AliasCombo.Text :=
ReadString(QueryName, 'Alias', '');
    NameEdit.Text :=
ReadString(QueryName, 'Name', '');
  end;
  Unmodify;
  if Showing then
    if NameEdit.Text <> '' then
      PasswordEdit.SetFocus else
      QueryEdit.SetFocus;
end;

<-------> procedure
TAdhocForm.SaveQueryAs;
begin
  if GetNewName(QueryName) then
  begin
    Unnamed := False;
    SaveQuery;
    with SavedQueryCombo, Items do
    begin
      if IndexOf(QueryName) < 0 then
Add(QueryName);
      ItemIndex :=
IndexOf(QueryName);
    end;
  end;
end;

<-------> procedure
TAdhocForm.SaveQuery;
begin
  if Unnamed then
    SaveQueryAs
```

```
  while S^ <> #0 do
    if (S[0] = '\') and (S[1] = 'n') then
    begin
      B^ := #13;
      Inc(B);
      B^ := #10;
      Inc(B);
      Inc(S);
      Inc(S);
    end
    else
    begin
      B^ := S^;
      Inc(B);
      Inc(S);
    end;
  B^ := #0;
  Result := Buffer;
end;

function TAdhocForm.IsModified:
Boolean;
begin
  Result := (AliasCombo.Text <>
OldAlias) or NameEdit.Modified or
    QueryEdit.Modified;
end;

function
TAdhocForm.CheckModified:
Boolean;
begin
  Result := True;
  if IsModified then
    case MessageDlg(Format('Query
%s modified, save?', [QueryName]),
      mtConfirmation,
mbYesNoCancel, 0) of
      mrYes: SaveQuery;
      mrCancel: Result := False;
    end;
end;

<-------> procedure
TAdhocForm.Unmodify;
begin
  OldAlias := AliasCombo.Text;
  NameEdit.Modified := False;
```

```
'Alias', 'IBLOCAL');

WriteString(VeryInefficientName,
'Name', 'SYSDBA');

SavedQueryCombo.Items.Add(Very
InefficientName);
     WriteString(AmountDueName,
'Query', AmountDueByCustomer);
     WriteString(AmountDueName,
'Alias', 'DBDEMOS');
     WriteString(AmountDueName,
'Name', '');

SavedQueryCombo.Items.Add(Amo
untDueName);
    end;
  end;


begin
  { Grab session aliases }

Session.GetAliasNames(AliasCombo
.Items);

  { Load in saved queries }
  SavedQueries :=
TIniFile.Create('BKQUERY.INI');

SavedQueries.ReadSections(SavedQ
ueryCombo.Items);
  if SavedQueryCombo.Items.Count
<= 0 then CreateInitialIni;
  SavedQueryCombo.ItemIndex := 0;
  QueryName :=
SavedQueryCombo.Items[0];
  Unmodify;
  ReadQuery;
end;

<-------> procedure
TAdhocForm.FormDestroy(Sender:
TObject);
begin
  SavedQueries.Free;
end;

<-------> procedure
```

```
  else
    with SavedQueries do
    begin
      WriteString(QueryName,
'Query',
StrToIniStr(QueryEdit.Text));
      WriteString(QueryName, 'Alias',
AliasCombo.Text);
      WriteString(QueryName,
'Name', NameEdit.Text);
      Unmodify;
    end;
end;

<-------> procedure
TAdhocForm.FormCreate(Sender:
TObject);
  <-------> procedure
CreateInitialIni;
  const
    VeryInefficientName = 'IB: Very
Inefficient Query';
    VeryInefficientQuery =
      'select EMP_NO, Avg(Salary) as
Salary\n'+
      '  from employee, employee,
employee\n' +
      '  group by EMP_NO';
    AmountDueName = 'DB: Amount
Due By Customer';
    AmountDueByCustomer =
      'select Company,
Sum(ItemsTotal) -
Sum(AmountPaid) as AmountDue\n'
+
      '  from customer, orders\n' +
      '  where Customer.CustNo =
Orders.CustNo\n' +
      '  group by Company';
  begin
    { Create initial INI file when one
doesn't already exisit }
    with SavedQueries do
    begin

WriteString(VeryInefficientName,
'Query', VeryInefficientQuery);

WriteString(VeryInefficientName,
```

```
TObject);
begin
  SaveQuery;
end;

<-------> procedure
TAdhocForm.SaveAsBtnClick(Send
er: TObject);
begin
  SaveQueryAs;
end;

<-------> procedure
TAdhocForm.SavedQueryComboCh
ange(Sender: TObject);
begin
  ReadQuery;
end;

<-------> procedure
TAdhocForm.FormCloseQuery(Sen
der: TObject;
  var CanClose: Boolean);
begin
  CanClose := CheckModified;
end;

end.


\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
2
unit ResltFrm;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  ExtCtrls, DB, DBTables, Grids,
DBGrids, StdCtrls;

type
  TQueryForm = class(TForm)
    QueryLabel: TLabel;
    DBGrid1: TDBGrid;
    DataSource: TDataSource;
```

```
TAdhocForm.CloseBtnClick(Sender
: TObject);
begin
  Close;
end;

<-------> procedure
TAdhocForm.ExecuteBtnClick(Send
er: TObject);
begin
  BackgroundQuery(QueryName,
AliasCombo.Text, NameEdit.Text,
PasswordEdit.Text,
    QueryEdit.Text);
  BringToFront;
end;

<-------> procedure
TAdhocForm.NewBtnClick(Sender:
TObject);

  function UniqueName: string;
  var
    I: Integer;
  begin
    I := 1;
    repeat
      Result := Format('Query%d',
[I]);
      Inc(I);
    until
SavedQueryCombo.Items.IndexOf(
Result) < 0;
  end;

begin
  AliasCombo.Text := 'DBDEMOS';
  NameEdit.Text := '';
  PasswordEdit.Text := '';
  QueryEdit.Text := '';
  QueryEdit.SetFocus;
  QueryName := UniqueName;
  SavedQueryCombo.ItemIndex := -
1;
  Unnamed := True;
end;

<-------> procedure
TAdhocForm.SaveBtnClick(Sender:
```

```
  Guard: Integer;
  Numbers: Integer;

{ Thread safe increment of Numbers to guarantee the result is unique }

function GetUniqueNumber: Integer;
asm
@@1:   MOV    EDX,1
       XCHG   Guard,EDX
       OR     EDX,EDX
       JNZ    @@2
       MOV    EAX,Numbers
       INC    EAX
       MOV    Numbers,EAX
       MOV    Guard,EDX
       RET

@@2:   PUSH   0
       CALL   Sleep
       JMP    @@1
end;


<-------> procedure TQueryThread.Execute;
var
  UniqueNumber: Integer;
begin
  try
    with QueryForm do
    begin
      { Ensure the Query has a unique session and database.  A unique session
        is required for each thread. Since databases are session specific
        it must be unique as well }
      UniqueNumber := GetUniqueNumber;
      Session.SessionName := Format('%s%x', [Session.Name, UniqueNumber]);
      Database.SessionName := Session.SessionName;

      Database.DatabaseName := Format('%s%x', [Database.Name,
```

```
    Query: TQuery;
    Session: TSession;
    StatusLine: TLabel;
    Database: TDatabase;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

<-------> procedure BackgroundQuery(const QueryName, Alias, User, Password,
  QueryText: string);

implementation

{$R *.dfm}

{ TQueryThread }

type
  TQueryThread = class(TThread)
  private
    QueryForm: TQueryForm;
    MessageText: string;
    <-------> procedure ConnectQuery;
    <-------> procedure DisplayMessage;
  protected
    <-------> procedure Execute; override;
  public
    constructor Create(AQueryForm: TQueryForm);
  end;

constructor TQueryThread.Create(AQueryForm: TQueryForm);
begin
  QueryForm := AQueryForm;
  FreeOnTerminate := True;
  inherited Create(False);
end;

var
```

```
  with QueryForm do StatusLine.Caption := MessageText;
end;

{ BackgroundQuery }

<-------> procedure BackgroundQuery(const QueryName, Alias, User, Password,
  QueryText: string);
var
  QueryForm: TQueryForm;
begin
  QueryForm := TQueryForm.Create(Application);
  with QueryForm, Database do
  begin
    Caption := QueryName;
    QueryLabel.Caption := QueryText;
    Show;
    AliasName := Alias;
    Params.Values['USER'] := User;
    Params.Values['PASSWORD'] := Password;
    Query.Sql.Text := QueryText;
  end;

  { Create the background thread to execute the query. Since the thread
    will free itself on termination we do not neet to maintain a reference
    to it.  Creating it is enough }
  TQueryThread.Create(QueryForm);
end;

end.

\\\\\\\\\\\\\\\\\\\\
3
unit SaveQAs;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
```

```
UniqueNumber]);
      Query.SessionName := Database.SessionName;
      Query.DatabaseName := Database.DatabaseName;

      { Open the query }
      Query.Open;

      { Connect the query to the grid. This must be done in a synchronzied
        method since assigning the query to the DataSource will modify the
        contents of the grid and the grid can only be modified from the main
        VCL thread }
      Synchronize(ConnectQuery);

      { Update the status line.  Since the label is a VCL control this must
        also be done in the main VCL thread }
      MessageText := 'Query openned';
      Synchronize(DisplayMessage);
    end;
  except
    on E: Exception do
    begin
      { Display any error we receive on the status line }
      MessageText := Format('%s: %s.', [E.ClassName, E.Message]);
      Synchronize(DisplayMessage);
    end;
  end;
end;

<-------> procedure TQueryThread.ConnectQuery;
begin
  with QueryForm do DataSource.Dataset := Query;
end;

<-------> procedure TQueryThread.DisplayMessage;
begin
```

القسم السادس

برمجة الألعاب

في لغة البرمجة دلفي

الملف التنفيذي لبرنامج
24 procedures
Swat)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
unit Main;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  ExtCtrls, Menus, StdCtrls;

const
  crMaletUp : integer = 5;
  crMaletDown : integer  = 6;
  MissedPoints : integer  = -2;
  HitPoints  : integer = 5;
  MissedCritter : integer = -1;
  CritterSize : integer = 72;
  TimerId  : integer = 1;

type
  THole = record
    Time : integer;
```

```
  StdCtrls, ExtCtrls;

type
  TSaveQueryAs = class(TForm)
    NameEdit: TEdit;
    Label1: TLabel;
    Bevel1: TBevel;
    OKBtn: TButton;
    CancelBtn: TButton;
    <-------> procedure
NameEditChange(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

function GetNewName(var
QueryName: string): Boolean;

implementation

{$R *.dfm}

function GetNewName(var
QueryName: string): Boolean;
begin
  with
TSaveQueryAs.Create(Application)
do
  begin
    NameEdit.Text := QueryName;
    Result := ShowModal = mrOK;
    if Result then QueryName :=
NameEdit.Text;
  end;
end;

<-------> procedure
TSaveQueryAs.NameEditChange(Se
nder: TObject);
begin
  OkBtn.Enabled := NameEdit.Text
<> '';
end;

end.
\\\\\\\\\\\\\\\\\\\\
```

```
    Dead : boolean;
  end;

  TSwatForm = class(TForm)
    MainMenu1: TMainMenu;
    Gamr1: TMenuItem;
    New1: TMenuItem;
    Options1: TMenuItem;
    Stop1: TMenuItem;
    Pause1: TMenuItem;
    About1: TMenuItem;
    Timer1: TTimer;
    GameOverImage: TImage;
    Image1: TImage;
    TimeLabel: TLabel;
    MissLabel: TLabel;
    HitsLabel: TLabel;
    EscapedLabel: TLabel;
    ScoreLabel: TLabel;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
Timer1Timer(Sender: TObject);
    <-------> procedure
FormMouseDown(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y:
Integer);
    <-------> procedure
FormMouseUp(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y:
Integer);
    <-------> procedure
New1Click(Sender: TObject);
    <-------> procedure
Options1Click(Sender: TObject);
    <-------> procedure
Stop1Click(Sender: TObject);
    <-------> procedure
Pause1Click(Sender: TObject);
    <-------> procedure
About1Click(Sender: TObject);
  private
    { Private declarations }
    Score : integer;
    Hits, Miss, Escaped : integer;
    IsGameOver, IsPause : Boolean;
    Live : TBitmap;
```

```
    Dead : TBitmap;
    HoleInfo : array[0..4] of THole;
    Holes : array[0..4] of TPoint;
    <-------> procedure WriteScore;
  public
    { Public declarations }
    LiveTime,  Frequence, GameTime
: integer;
  end;

var
  SwatForm: TSwatForm;

implementation

uses options, about;

{$R *.dfm}
{$R extrares.res}

<-------> procedure
TSwatForm.FormCreate(Sender:
TObject);
begin
  Holes[0] := Point( 10, 10 );
  Holes[1] := Point( 200, 10 );
  Holes[2] := Point( 100, 100 );
  Holes[3] := Point( 10, 200 );
  Holes[4] := Point( 200, 200 );

  Screen.Cursors[crMaletUp] :=
LoadCursor(HInstance, 'Malet');
  Screen.Cursors[crMaletDown] :=
LoadCursor(HInstance,
'MaletDown');
  Screen.Cursor :=
TCursor(crMaletUp);

  randomize;

  Live := TBitmap.Create;

Live.LoadFromResourceName(HIns
tance, 'Live');
  Dead := TBitmap.Create;

Dead.LoadFromResourceName(HIn
stance, 'Dead');

```

```
  WriteScore;
  if (Timer1.Tag >= GameTime) then
    Stop1Click(self);
end;

<-------> procedure TSwatForm.FormMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  i : integer;
  hit : boolean;
begin
  Screen.Cursor := TCursor(crMaletDown);

  if (IsGameOver or IsPause) then
    exit;

  hit := false;
  for i := 0 to 4 do
    if ( (not HoleInfo[i].Dead) and (HoleInfo[i].Time <> 0) ) then
      if (X > Holes[i].x ) and ( X < (Holes[i].x + Live.Width) ) and
         ( Y > Holes[i].y ) and ( Y < (Holes[i].y + Live.Height)) then
      begin
        inc( Score, HitPoints );
        HoleInfo[i].Dead := true;
        HoleInfo[i].Time := Timer1.Tag + 2 * LiveTime;
        inc( Hits );
        hit := true;
        Canvas.Draw(Holes[i].x, Holes[i].y, Dead);
      end;
  if not(hit) then
  begin
    inc ( Score, MissedPoints );
    inc( Miss );
  end;
  WriteScore;
end;

<-------> procedure TSwatForm.FormMouseUp(Sender: TObject; Button: TMouseButton;
```

```
  IsGameOver := true;
  IsPause := false;
  LiveTime := 10;
  Frequence := 20;
  GameTime := 150;       // fifteen seconds

  Application.OnMinimize := Pause1Click;
  Application.OnRestore := Pause1Click;
end;

<-------> procedure TSwatForm.Timer1Timer(Sender: TObject);
var
  i : integer;
begin
  Timer1.Tag := Timer1.Tag + 1;
  i := random(Frequence);
  if (i < 5) then
  begin
    if (HoleInfo[i].Time = 0) then
    begin
      HoleInfo[i].Time := Timer1.Tag + LiveTime;
      HoleInfo[i].Dead := false;
      Canvas.Draw(Holes[i].x, Holes[i].y, Live);
    end;
  end;
  for i := 0 to 4 do
  begin
    if ( (Timer1.Tag > HoleInfo[i].Time ) and ( HoleInfo[i].Time <> 0 ) ) then
    begin
      HoleInfo[i].Time := 0;
      if not(HoleInfo[i].Dead) then
      begin
        inc( Score, MissedCritter );
        inc( Escaped );
      end;
      Canvas.FillRect(Rect(Holes[i].x, Holes[i].y, Holes[i].x + Dead.Width, Holes[i].y + Dead.Height));
    end;
  end;
```

```
  New1.Enabled := true;
  Options1.Enabled := true;
  Stop1.Enabled := false;
  for i := 0 to 4 do
    if (HoleInfo[i].Time <> 0) then
      Canvas.FillRect(Rect(Holes[i].x,
Holes[i].y, Holes[i].x + Dead.Width,
        Holes[i].y + Dead.Height));
end;

<-------> procedure
TSwatForm.Pause1Click(Sender:
TObject);
begin
  if (IsGameOver) then
    exit;

  if (IsPause) then
  begin
    IsPause := false;
    Pause1.Caption := '&Pause';
    Stop1.Enabled := true;
    Timer1.Enabled := true;
  end
  else
  begin
    IsPause := true;
    Pause1.Caption := '&Continue';
    Stop1.Enabled := false;
    Timer1.Enabled := false;
  end;
end;

<-------> procedure
TSwatForm.About1Click(Sender:
TObject);
begin
  AboutBox.ShowModal;
end;

<-------> procedure
TSwatForm.WriteScore;
begin
  TimeLabel.Caption :=
IntToStr(GameTime - Timer1.Tag);
  HitsLabel.Caption :=
IntToStr(Hits);
  MissLabel.Caption :=
```

```
  Shift: TShiftState; X, Y: Integer);
begin
  Screen.Cursor :=
TCursor(crMaletUp);
end;

<-------> procedure
TSwatForm.New1Click(Sender:
TObject);
begin
  Timer1.Enabled := true;
  Timer1.Tag := 0;
  Score := 0;
  Hits := 0;
  Miss := 0;
  Escaped := 0;
  if (IsPause)
  then begin
    IsPause := false;
    Pause1.Caption := '&Pause';
  end;
  GameOverImage.Visible := false;
  IsGameOver := false;
  FillChar(HoleInfo,
sizeof(HoleInfo), 0);
  New1.Enabled := false;
  Options1.Enabled := false;
  Stop1.Enabled := true;
end;

<-------> procedure
TSwatForm.Options1Click(Sender:
TObject);
begin
  OptionsDlg.ShowModal;
end;

<-------> procedure
TSwatForm.Stop1Click(Sender:
TObject);
var
 i : integer;
begin
  Timer1.Enabled := false;
  IsPause := false;
  GameOverImage.Visible := true;
  IsGameOver := true;
  Timer1.Tag := GameTime;
```

```
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
3
unit options;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls, ComCtrls, ExtCtrls;

type
  TOptionsDlg = class(TForm)
    Bevel1: TBevel;
    Slow: TLabel;
    Fast: TLabel;
    Label1: TLabel;
    Label2: TLabel;
    Speed: TLabel;
    Population: TLabel;
    Time: TLabel;
    OKBtn: TButton;
    CancelBtn: TButton;
    SpeedSet: TTrackBar;
    PopulationSet: TTrackBar;
    GameTimeSet: TEdit;
    <-------> procedure
OKBtnClick(Sender: TObject);
    <-------> procedure
FormShow(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  OptionsDlg: TOptionsDlg;

implementation

uses Main;

{$R *.dfm}

<-------> procedure
TOptionsDlg.OKBtnClick(Sender:
```

```
IntToStr(Miss);
  EscapedLabel.Caption :=
IntToStr(Escaped);
  ScoreLabel.Caption :=
IntToStr(Score);
end;

end.

\\\\\\\\\\\\\\\\\\
2
unit about;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls, ExtCtrls;

type
  TAboutBox = class(TForm)
    Panel1: TPanel;
    ProgramIcon: TImage;
    ProductName: TLabel;
    Hits: TLabel;
    Copyright: TLabel;
    Comments: TLabel;
    Escape: TLabel;
    Miss: TLabel;
    Label1: TLabel;
    OKButton: TButton;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  AboutBox: TAboutBox;

implementation

{$R *.dfm}

end.
```

```
TObject);
begin
  SwatForm.LiveTime :=
SpeedSet.Max + 1 -
SpeedSet.Position;
  SwatForm.Frequence :=
PopulationSet.Position;
  SwatForm.GameTime :=
StrToInt(GameTimeSet.Text);

  // limit the value of GameTime to a
reasonable length
  if (SwatForm.GameTime < 1) then
    SwatForm.GameTime := 150;
  if (SwatForm.GameTime > 9999)
then
    SwatForm.GameTime := 9999;
end;

<-------> procedure
TOptionsDlg.FormShow(Sender:
TObject);
begin
  SpeedSet.Position := SpeedSet.Max
+ 1 - SwatForm.LiveTime;
  PopulationSet.Position :=
SwatForm.Frequence;
  GameTimeSet.Text :=
inttoStr(SwatForm.GameTime);
end;

end.
\\\\\\\\\\\\\\\\\\
```

<-------> object s41

الملف النصي لبرنامج

Swat)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<-------> object SwatForm:
TSwatForm
  Left = 570
  Top = 159
  Width = 308
  Height = 494
  Caption = 'Swat!'
  Color = clSilver
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  Menu = MainMenu1
  OldCreateOrder = False
  OnCreate = FormCreate
  OnMouseDown =
FormMouseDown
```

```
    Top = 351
    Width = 9
    Height = 19
    Caption = '0'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -16
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    Transparent = True
  end
  <-------> object EscapedLabel: TLabel
    Left = 134
    Top = 351
    Width = 9
    Height = 19
    Caption = '0'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -16
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    Transparent = True
  end
  <-------> object ScoreLabel: TLabel
    Left = 205
    Top = 351
    Width = 9
    Height = 19
    Caption = '0'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -16
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    Transparent = True
  end
  <-------> object MainMenu1: TMainMenu
    <-------> object Gamr1: TMenuItem
```

```
  OnMouseUp = FormMouseUp
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object GameOverImage: TImage
    Left = 10
    Top = 72
    Width = 262
    Height = 145
    AutoSize = True
    Picture.Data = {

8888888888888888888888888888880000}
    Transparent = True
  end
  <-------> object TimeLabel: TLabel
    Left = 122
    Top = 296
    Width = 18
    Height = 19
    Caption = '00'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -16
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    Transparent = True
  end
  <-------> object MissLabel: TLabel
    Left = 78
    Top = 351
    Width = 9
    Height = 19
    Caption = '0'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -16
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    Transparent = True
  end
  <-------> object HitsLabel: TLabel
    Left = 41
```

Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = False
Position = poScreenCenter
OnShow = FormShow
PixelsPerInch = 96
TextHeight = 13
<-------> object Bevel1: TBevel
Left = 8
Top = 8
Width = 281
Height = 217
Shape = bsFrame
end
<-------> object Slow: TLabel
Left = 36
Top = 72
Width = 23
Height = 13
Caption = 'Slow'
end
<-------> object Fast: TLabel
Left = 241
Top = 72
Width = 20
Height = 13
Caption = 'Fast'
end
<-------> object Label1: TLabel
Left = 36
Top = 144
Width = 20
Height = 13
Caption = 'Low'
end
<-------> object Label2: TLabel
Left = 239
Top = 144
Width = 22
Height = 13
Caption = 'High'
end
<-------> object Speed: TLabel
Left = 133

Caption = '&Game'
<-------> object New1: TMenuItem
Caption = '&New'
OnClick = New1Click
end
<-------> object Options1: TMenuItem
Caption = '&Options...'
OnClick = Options1Click
end
<-------> object Stop1: TMenuItem
Caption = '&Stop'
Enabled = False
OnClick = Stop1Click
end
end
<-------> object Pause1: TMenuItem
Caption = '&Pause'
OnClick = Pause1Click
end
<-------> object About1: TMenuItem
Caption = '&About'
OnClick = About1Click
end
end
<-------> object Timer1: TTimer
Enabled = False
Interval = 100
OnTimer = Timer1Timer
Left = 32
end
End
\\\\\\\\\\\\\\\\\\\\
2
<-------> object OptionsDlg: TOptionsDlg
Left = 174
Top = 188
BorderStyle = bsDialog
Caption = 'Options'
ClientHeight = 233
ClientWidth = 384
Color = clBtnFace

Height = 25
Max = 30
TabOrder = 2
end
<-------> object PopulationSet: TTrackBar
Left = 36
Top = 112
Width = 225
Height = 25
Max = 35
TabOrder = 3
end
<-------> object GameTimeSet: TEdit
Left = 116
Top = 184
Width = 65
Height = 21
TabOrder = 4
Text = '150'
end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\
3
<-------> object AboutBox: TAboutBox
Left = 150
Top = 135
BorderStyle = bsDialog
Caption = 'About Swat!...'
ClientHeight = 213
ClientWidth = 209
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = False
Position = poScreenCenter
PixelsPerInch = 96
TextHeight = 13
<-------> object Panel1: TPanel
Left = 8
Top = 8

Top = 24
Width = 31
Height = 13
Caption = '&Speed'
FocusControl = SpeedSet
end
<-------> object Population: TLabel
Left = 123
Top = 96
Width = 50
Height = 13
Caption = '&Population'
FocusControl = PopulationSet
end
<-------> object Time: TLabel
Left = 121
Top = 168
Width = 54
Height = 13
Caption = '&Game Time'
FocusControl = GameTimeSet
end
<-------> object OKBtn: TButton
Left = 300
Top = 8
Width = 75
Height = 25
Caption = 'OK'
Default = True
ModalResult = 1
TabOrder = 0
OnClick = OKBtnClick
end
<-------> object CancelBtn: TButton
Left = 300
Top = 38
Width = 75
Height = 25
Cancel = True
Caption = 'Cancel'
ModalResult = 2
TabOrder = 1
end
<-------> object SpeedSet: TTrackBar
Left = 36
Top = 40
Width = 225

```
    Left = 61
    Top = 104
    Width = 73
    Height = 13
    Alignment = taCenter
    Caption = 'Delphi Example'
    WordWrap = True
    IsControl = True
  end
  <-------> object Escape: TLabel
    Left = 55
    Top = 72
    Width = 82
    Height = 13
    Alignment = taCenter
    Caption = '-1 pts per escape'
    IsControl = True
  end
  <-------> object Miss: TLabel
    Left = 63
    Top = 88
    Width = 67
    Height = 13
    Alignment = taCenter
    Caption = '-2 pts per miss'
    IsControl = True
  end
  <-------> object Label1: TLabel
    Left = 48
    Top = 136
    Width = 97
    Height = 13
    Alignment = taCenter
    Caption = 'Borland
International'
    IsControl = True
  end
 end
 <-------> object OKButton: TButton
  Left = 67
  Top = 180
  Width = 75
  Height = 25
  Caption = 'OK'
  Default = True
  ModalResult = 1
  TabOrder = 1
 end
  Width = 193
  Height = 161
  BevelInner = bvRaised
  BevelOuter = bvLowered
  Caption = 'Panel1'
  ParentColor = True
  TabOrder = 0
  <-------> object ProgramIcon:
TImage
    Left = 80
    Top = 24
    Width = 33
    Height = 33
    Picture.Data = {

     E0FC0003}
    Stretch = True
    IsControl = True
  end
  <-------> object ProductName:
TLabel
    Left = 83
    Top = 8
    Width = 27
    Height = 13
    Caption = 'Swat!'
    IsControl = True
  end
  <-------> object Hits: TLabel
    Left = 69
    Top = 56
    Width = 55
    Height = 13
    Alignment = taCenter
    Caption = '5 pts per hit'
    IsControl = True
  end
  <-------> object Copyright: TLabel
    Left = 55
    Top = 120
    Width = 83
    Height = 13
    Alignment = taCenter
    Caption = 'Copyright © 1998'
    IsControl = True
  end
  <-------> object Comments:
TLabel
```

```
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

الملف التنفيذي لبرنامج

12 procedures

EarthPng)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit Main;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs, ExtCtrls;

type
  TForm1 = class(TForm)
    <-------> procedure
FormPaint(Sender: TObject);
    <-------> procedure
FormActivate(Sender: TObject);
    <-------> procedure
FormMouseMove(Sender: TObject;
Shift: TShiftState; X, Y: Integer);
    <-------> procedure
FormCreate(Sender: TObject);
  private
    { Private declarations }
    FGameOver : Boolean;
  public
    { Public declarations }
    backgroundImage : TImage;
    spriteImage : TImage;
    paddle : TImage;
    backgroundCanvas : TCanvas;
    workCanvas : TCanvas;
    backgroundRect, spriteRect,
changeRect, paddleRect,
changePaddleRect :TRect;
    x, y, xDir, yDir, paddleX,
paddleY, paddleCenter, Angle :
integer;
    <-------> procedure IdleLoop(
Sender: TObject; var Done: Boolean
);
    <-------> procedure
WMSetCursor(var Message:
TWMSetCursor); message
WM_SETCURSOR;
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

uses MMSystem;

<-------> procedure
TForm1.FormPaint(Sender:
TObject);
begin

RealizePalette(backgroundCanvas.H
andle);

RealizePalette(workCanvas.Handle);

Canvas.CopyRect(backgroundRect,
workCanvas, backgroundRect);
```

```
end;

<-------> procedure TForm1.FormActivate(Sender: TObject);
var
  backgrounddc, workdc : HDC;
  bkbmp, bmp : HBITMAP;
begin
  backgroundImage := TImage.Create( Self );
  spriteImage := TImage.Create( Self );
  paddle := TImage.Create( Self );
  workCanvas := TCanvas.Create;
  backgroundCanvas := TCanvas.Create;

  Angle := 1;

  spriteImage.Picture.LoadFromFile('Earth.ico');
  backgroundImage.Picture.LoadFromFile('androm.bmp');
  paddle.Picture.LoadFromFile('paddle.ico');

  WindowState := wsMaximized;

  backgroundRect.Top := 0;
  backgroundRect.Left := 0;
  backgroundRect.Right := ClientWidth;
  backgroundRect.Bottom := ClientHeight;


  spriteRect.Top := 0;
  spriteRect.Left := 0;
  spriteRect.Right := spriteImage.Picture.Width;
  spriteRect.Bottom := spriteImage.Picture.Height;

  //Set up backgroundCanvas
  backgrounddc := CreateCompatibleDC(Canvas.Handle);
  bkbmp := CreateCompatibleBitmap(Canvas.Handle, ClientWidth, ClientHeight);
  SelectObject(backgrounddc, bkbmp);
  SelectPalette(backgrounddc, backgroundImage.Picture.Bitmap.Palette, false);
  backgroundCanvas.Handle := backgrounddc;
  backgroundCanvas.StretchDraw( backgroundRect, backgroundImage.Picture.Bitmap);


  //Set up workCanvas
  workdc := CreateCompatibleDC(Canvas.Handle);
  bmp := CreateCompatibleBitmap(Canvas.Handle, ClientWidth, ClientHeight);
  SelectObject(workdc, bmp);
  SelectPalette(workdc, backgroundImage.Picture.Bitmap.Palette, false);
  workCanvas.Handle := workdc;

  workCanvas.CopyRect(backgroundRect,  backgroundCanvas, backgroundRect);
  workCanvas.Draw( 0, 0, spriteImage.Picture.Icon);
  paddleX := ClientWidth div 2;
  paddleY := ClientHeight - 50;
  workCanvas.Draw( paddleX, paddleY, paddle.Picture.Icon);

  paddleRect.Left := paddleX - paddle.Width;
  paddleRect.Right := paddleX + paddle.Width;
  paddleRect.Top       := paddleY;
  paddleRect.Bottom := paddleY + paddle.Height;
```

```
  else
    Application.Terminate;
end;

<-------> procedure
TForm1.IdleLoop(Sender: TObject;
var Done: Boolean);
var
  choice, SideDef, TopDef,
PaddleDifference: integer;
begin
  //keeps loop going
  done := false;
  //slows down action
  Sleep(1);

  changeRect := spriteRect;
  spriteRect.Left := x;
  spriteRect.Top := y;
  spriteRect.Right := x +
spriteImage.Picture.Width;
  spriteRect.Bottom := y +
spriteImage.Picture.Height;

  workCanvas.CopyRect(paddleRect,
backgroundCanvas, paddleRect);

  changePaddleRect := paddleRect;
  paddleRect.Left := paddleCenter -
((paddle.Picture.Width) div 2);
  paddleX := paddleRect.Left;
  paddleRect.Top := paddleY;
  paddleRect.Right := paddleX +
paddle.Picture.Width;
  paddleRect.Bottom := paddleY +
paddle.Picture.Height;

  SideDef := changeRect.Left -
spriteRect.Left;
  // If SideDiff < 0 the paddle is to the
right
  if(SideDef < 0) then
  begin
    changeRect.Right :=
spriteRect.Right;
  end
  else
  begin
    changeRect.Left :=
```

```
RealizePalette(backgroundCanvas.H
andle);

RealizePalette(workCanvas.Handle);

Canvas.CopyRect(backgroundRect,
workCanvas, backgroundRect);
end;

<-------> procedure
TForm1.FormMouseMove(Sender:
TObject; Shift: TShiftState; X,
  Y: Integer);
begin
  //Animates and moves paddle
  paddleCenter := X;
  if(paddleCenter <
paddle.Picture.Width div 2) then
      paddleCenter :=
paddle.Picture.Width div 2;
  if(paddleCenter > ClientWidth -
(paddle.Picture.Width div 2)) then
      paddleCenter := ClientWidth -
(paddle.Picture.Width div 2);
end;

<-------> procedure
TForm1.FormCreate(Sender:
TObject);
begin
  //Assign idle time function
  Application.OnIdle := IdleLoop;

  if(Application.MessageBox('Would
you like to play with Earth?', 'Hello
Earthling', MB_OKCANCEL) =
IDOK) then
  begin
      //load sound effect
      sndPlaySound('Utopia
Default.wav', SND_ASYNC or
SND_FILENAME);
      x := 0;
      y := 0;
      FGameOver := false;
      ShowCursor(false);

  end
```

```
  and ( (spriteRect.Right) >=
(paddleRect.Left) )
  and ( (spriteRect.Left) <=
(paddleRect.Right) ) then
  begin
    yDir := -5;
    sndPlaySound('Utopia
Default.wav', SND_ASYNC or
SND_FILENAME);
  end;

  if (x <= 0) then
  begin
    xDir := 5;
  end;

  if(x >= ClientWidth - 16) then
  begin
    xDir := -5;
  end;

  inc ( x , xDir );
  inc ( y , yDir );

  PaddleDifference :=
changePaddleRect.Left -
paddleRect.Left;
  // If PaddleDiff < 0 the paddle is to
the right
  if(PaddleDifference < 0) then
  begin
    changePaddleRect.Right :=
paddleRect.Right;
  end
  else
  begin
    changePaddleRect.Left :=
paddleRect.Left;
  end;

  //Perform dirty rectangle animation
on memory and Form canvas
  workCanvas.Draw(x, y,
spriteImage.Picture.Icon);
  workCanvas.Draw(paddleX,
paddleY, paddle.Picture.Icon);

RealizePalette(backgroundCanvas.H
andle);
spriteRect.Left;
  end;

  TopDef := changeRect.Top -
spriteRect.Top;
  // If SideDiff < 0 the paddle is to the
Down
  if(TopDef < 0) then
  begin
    changeRect.Bottom :=
spriteRect.Bottom;
  end
  else
  begin
    changeRect.Top :=
spriteRect.Top;
  end;

  workCanvas.CopyRect(spriteRect,
backgroundCanvas, spriteRect);
  //ChangeRectCalcs
  if (y <= 0) then
  begin
    yDir := 5;
  end;
  if (y >= ClientHeight - 16) then
  begin
    FGameOver := true;
    SetCursor(HCURSOR(
IDC_ARROW ));
    ShowCursor(true);
    choice := MessageBox(Handle,
'You lost Earth', 'Try Again?',
MB_RETRYCANCEL);
    if(choice = IDRETRY) then
    begin
      x := 0;
      y := 0;
      ShowCursor(false);
    end
    else
      Form1.Close;
  end;

  if ( (spriteRect.Bottom - 16) >=
(paddleRect.Top) )
   and ( (spriteRect.Bottom - 16) <=
(paddleRect.Top + 5) )
```

```
  OnActivate = FormActivate
  OnCreate = FormCreate
  OnMouseMove = FormMouseMove
  OnPaint = FormPaint
  PixelsPerInch = 96
  TextHeight = 13
end
```

<-------> object 5

الملف النصي لبرنامج

Football)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<-------> object AboutForm: TAboutForm
  Left = 422
  Top = 149
  Width = 237
  Height = 416
  Caption = 'About Football...'
  Color = clBlue
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  Position = poScreenCenter
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Label1: TLabel
    Left = 29
    Top = 6
    Width = 171
    Height = 16
    Caption = 'FOOTBALL INSTRUCTIONS'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWhite
```

```
RealizePalette(workCanvas.Handle);
  Canvas.CopyRect(changeRect, workCanvas, changeRect);

Canvas.CopyRect(changePaddleRect, workCanvas, changePaddleRect);
end;

<-------> procedure TForm1.WMSetCursor(var Message: TWMSetCursor);
begin
  //Hides Cursor
  if not(FGameOver) then
  begin
    SetCursor( HCURSOR( nil ) );
  end;
end;

end.
```

<-------> object 1

الملف النصي لبرنامج

EarthPng)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
<-------> object Form1: TForm1
  Left = 111
  Top = 112
  Width = 544
  Height = 375
  Caption = 'Earth Pong'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  Icon.Data = {
   }
  OldCreateOrder = False
```

```
one.'
    '(Just kidding.)')
  ParentCtl3D = False
  ParentFont = False
  ReadOnly = True
  TabOrder = 0
 end
 <-------> object Button1: TButton
  Left = 77
  Top = 355
  Width = 75
  Height = 25
  Caption = 'OK'
  ModalResult = 1
  TabOrder = 1
 end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

الملف التنفيذي لبرنامج
40 procedures
Football)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
unit Main;

interface

uses
 Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
 Menus, ExtCtrls, StdCtrls,
ComCtrls, MMSystem;

const
 RBTOP : integer = 58;
 RBMIDDLE  : integer = 74;
 RBBOTTOM  : integer = 90;
 RBLEFT  : integer = 36;
 TACKLETOP  : integer = 59;
```

```
  Font.Height = -13
  Font.Name = 'Arial'
  Font.Style = [fsBold]
  ParentFont = False
 end
 <-------> object Memo1: TMemo
  Left = 18
  Top = 27
  Width = 192
  Height = 321
  BorderStyle = bsNone
  Color = clBlue
  Ctl3D = False
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWhite
  Font.Height = -11
  Font.Name = 'Default'
  Font.Style = []
  Lines.Strings = (
    'The Computer''s on Defense.'
    'You Control the Running Back.'
    ''
    'Slide OFF switch to PRO 1'
    '(PRO 2 for advanced players).'
    ''
    'Press ST button for downs, field'
    'position and yards to go.'
    ''
    'Press ARROW buttons (or
I,K,J,L'
    'keys) to maneuver running back
'
    '(bright blip) through tacklers '
    '(dimmer blips).  Tackled?'
    'Game stops.'
    ''
    'Press ST button to check status
and'
    'reset field.  Try another play.'
    ''
    'On fourth down, try a run, or
push'
    'K button to punt or kick field
goal.'
    ''
    'If the game malfunctions it may'
    'mean battery wear.  Use a fresh
```

About1: TMenuItem;
Clock: TTimer;
<-------> procedure About1Click(Sender: TObject);
<-------> procedure Exit1Click(Sender: TObject);
<-------> procedure FormKeyPress(Sender: TObject; var Key: Char);
<-------> procedure MoveUpClick(Sender: TObject);
<-------> procedure MoveDownClick(Sender: TObject);
<-------> procedure MoveForwardClick(Sender: TObject);
<-------> procedure MoveBackClick(Sender: TObject);
<-------> procedure ComputerClick(Sender: TObject);
<-------> procedure ClockTimer(Sender: TObject);
<-------> procedure TimerTimer(Sender: TObject);
<-------> procedure ScoreMouseUp(Sender: TObject; Button: TMouseButton;
Shift: TShiftState; X, Y: Integer);
<-------> procedure ScoreMouseDown(Sender: TObject; Button: TMouseButton;
Shift: TShiftState; X, Y: Integer);
<-------> procedure KickClick(Sender: TObject);
<-------> procedure OnOffSwitchChange(Sender: TObject);
private
{ Private declarations }
field : array[0..9]of array[0..2] of spot; // the X and Y coordinates of the field
rbx, rby : integer; // the X and Y coordinates of the runningback
running : boolean;
LastSack : TLabel;
YardsToGo : integer;

TACKLEMIDDLE : integer = 75;
TACKLEBOTTOM : integer = 91;
TACKLEROW1 : integer = 104;
TACKLEROW2 : integer = 148;
TACKLEROW3 : integer = 236;
type
spot = record
player : TLabel;
rb : boolean;
end;
TMainForm = class(TForm)
Image7: TImage;
HomeDisplay: TLabel;
TimeDisplay: TLabel;
VisitorDisplay: TLabel;
Runningback: TLabel;
Tackler1: TLabel;
Tackler2: TLabel;
Tackler3: TLabel;
Tackler4: TLabel;
Tackler5: TLabel;
HomeLabel: TLabel;
TimeLabel: TLabel;
VisitorLabel: TLabel;
MoveForward: TImage;
MoveUp: TImage;
MoveDown: TImage;
Kick: TImage;
Score: TImage;
Label2: TLabel;
Label1: TLabel;
Label3: TLabel;
DownDisplay: TLabel;
FieldPosDisplay: TLabel;
YTGDisplay: TLabel;
DownLabel: TLabel;
FieldPosLabel: TLabel;
YTGLabel: TLabel;
MoveBack: TImage;
OnOffSwitch: TTrackBar;
Computer: TButton;
Timer: TTimer;
MainMenu1: TMainMenu;
File1: TMenuItem;
Exit1: TMenuItem;
Help1: TMenuItem;

```
    'i':
      MoveUpClick(Sender);
    'k':
      MoveDownClick(Sender);
    'l':
      MoveForwardClick(Sender);
    'j':
      MoveBackClick(Sender);
  end;
end;

<-------> procedure
TMainForm.MoveUpClick(Sender:
TObject);
begin
  if ((OnOffSwitch.Position = 2) or
(LastSack <> Nil)) then
    exit;

  if not running then
    TogglePlay(true);

  if (Runningback.Top = RBTOP)
then
    exit; // already as up as it gets

  if (field[rbx][rby-1].player <> Nil)
then
  begin
    Sacked(field[rbx][rby-1].player);
    exit;
  end;

  if (Runningback.Top =
RBBOTTOM) then
    Runningback.Top := RBMIDDLE
  else if (Runningback.Top =
RBMIDDLE) then
    Runningback.Top := RBTOP;

  field[rbx][rby].player := Nil;
  field[rbx][rby].rb := false;
  dec( rby );
  field[rbx][rby].player :=
Runningback;
  field[rbx][rby].rb := true;
end;

```

```
    Down : integer;
    FieldPosition : integer;
    Home : integer;
    Visitor : integer;
    Quarter : integer;
    TimeLeft : double;
    <-------> procedure
ShowField(visible : boolean);
    <-------> procedure ResetField;
    <-------> procedure Sacked(Player
: TLabel);
    <-------> procedure
ShowDisplay(visible : boolean);
    <-------> procedure TogglePlay(
toggle : boolean );
    <-------> procedure ResetGame;
  public
    { Public declarations }
  end;

var
  MainForm: TMainForm;

implementation

uses about;

{$R *.dfm}

<-------> procedure
TMainForm.About1Click(Sender:
TObject);
begin
  AboutForm.ShowModal;
end;

<-------> procedure
TMainForm.Exit1Click(Sender:
TObject);
begin
  Application.Terminate;
end;

<-------> procedure
TMainForm.FormKeyPress(Sender:
TObject; var Key: Char);
begin
  case Key of
```

```
   TogglePlay(true);

  if (Runningback.Left <
TACKLEROW3 - 2) then
  begin
   if (field[rbx+1][rby].player <> Nil)
then
   begin
     Sacked(field[rbx+1][rby].player);
     exit;
   end;
   field[rbx][rby].player := Nil;
   field[rbx][rby].rb := false;
   Runningback.Left :=
Runningback.Left + 22;
   inc( rbx );
  end
  else
  begin
   if (field[0][rby].player <> Nil) then
   begin
      Sacked(field[0][rby].player);
      exit;
   end;
   field[rbx][rby].player := Nil;
   field[rbx][rby].rb := false;
   Runningback.Left := RBLEFT;
   rbx := 0;
  end;

  dec( YardsToGo );
  inc( FieldPosition );
  if (FieldPosition = 100) then
  begin
   sndPlaySound('td.wav',
SND_SYNC);
   running := false;
   ShowField(false);
   inc( Home, 7);
   HomeDisplay.Caption :=
inttostr(Home);
   FieldPosition:=80;  // Starting
position for visitor
   Down := 1;
   YardsToGo := 10;
   Computer.Visible := true;
   exit;
  end;
```

```
<-------> procedure
TMainForm.MoveDownClick(Sende
r: TObject);
begin
  if ((OnOffSwitch.Position = 2) or
(LastSack <> Nil)) then
     exit;

  if not running then
     TogglePlay(true);

  if (Runningback.Top =
RBBOTTOM) then
     exit; // already as down as it gets

  if (field[rbx][rby+1].player <> Nil)
then
  begin

Sacked(field[rbx][rby+1].player);
     exit;
  end;

  if (Runningback.Top = RBTOP)
then
     Runningback.Top :=
RBMIDDLE
  else if (Runningback.Top =
RBMIDDLE) then
     Runningback.Top :=
RBBOTTOM;

  field[rbx][rby].player := Nil;
  field[rbx][rby].rb := false;
  inc( rby );
  field[rbx][rby].player :=
Runningback;
  field[rbx][rby].rb := true;
end;

<-------> procedure
TMainForm.MoveForwardClick(Se
nder: TObject);
begin
 if ((OnOffSwitch.Position = 2) or
(LastSack <> Nil)) then
  exit;

 if not running then
```

```
  Inc( YardsToGo );
  Dec( FieldPosition );

  field[rbx][rby].player :=
Runningback;
  field[rbx][rby].rb := true;
 end;
end;

<-------> procedure
TMainForm.ResetField;
var
 x, y :integer;
begin
 TogglePlay(false);
 LastSack := Nil;

 // empty the field
 for y := 0 to 2 do
  for x := 0 to 9 do
  begin
    field[x][y].player := Nil;
    field[x][y].rb := false;
  end;

 // initial locations of players
 field[0][1].player := Runningback;
 field[0][1].rb := true;
 rbx := 0;
 rby := 1;
 field[3][0].player := Tackler1;
 field[3][1].player := Tackler2;
 field[3][2].player := Tackler3;
 field[5][1].player := Tackler4;
 field[9][1].player := Tackler5;

 Runningback.Left := RBLEFT;
 Runningback.Top := RBMIDDLE;
 Tackler1.Left := TACKLEROW1;
 Tackler1.Top := TACKLETOP;
 Tackler2.Left := TACKLEROW1;
 Tackler2.Top :=
TACKLEMIDDLE;
 Tackler3.Left := TACKLEROW1;
 Tackler3.Top :=
TACKLEBOTTOM;
 Tackler4.Left := TACKLEROW2;
 Tackler4.Top :=
```

```
  field[rbx][rby].player :=
Runningback;
  field[rbx][rby].rb := true;
end;

<-------> procedure
TMainForm.MoveBackClick(Sender
: TObject);
begin
 if ((OnOffSwitch.Position = 2) or
(LastSack <> Nil)) then
   exit;

 if not running then
   TogglePlay(true);
 if(FieldPosition > 0) then
 begin
   if (Runningback.Left > RBLEFT
+ 2) then
   begin
     if (field[rbx-1][rby].player <>
Nil) then
     begin
      Sacked(field[rbx-
1][rby].player);
      exit;
     end;
     field[rbx][rby].player := Nil;
     field[rbx][rby].rb := false;
     Runningback.Left :=
Runningback.Left - 22;
     dec( rbx );
   end
   else
   begin
     if (field[9][rby].player <> Nil)
then
     begin
      Sacked(field[9][rby].player);
      exit;
     end;
     field[rbx][rby].player := Nil;
     field[rbx][rby].rb := false;
     Runningback.Left :=
TACKLEROW3;
     rbx := 9;
   end;
```

```
<-------> procedure TMainForm.ShowField(visible: boolean);
begin
  Runningback.Visible := visible;
  Tackler1.Visible := visible;
  Tackler2.Visible := visible;
  Tackler3.Visible := visible;
  Tackler4.Visible := visible;
  Tackler5.Visible := visible;
end;

<-------> procedure TMainForm.TogglePlay(toggle: boolean);
begin
  running := toggle;
  Timer.Enabled := toggle;
  Clock.Enabled := toggle;
end;

<-------> procedure TMainForm.ComputerClick(Sender : TObject);
begin
  ShowField(false);
  Dec( FieldPosition, random(100) );
  if (FieldPosition <= 0) then
  begin
    sndPlaySound('td.wav', SND_ASYNC);
    inc( Visitor, 7);
    VisitorDisplay.Caption := IntToStr(Visitor);
    FieldPosition := 20;
  end
  else
  begin
    sndPlaySound('whistle.wav', SND_SYNC);
    sndPlaySound('whistle.wav', SND_SYNC);
  end;
  Computer.Visible := false;
  LastSack := Runningback; // hack to keep movement keys disabled
end;

<-------> procedure
TACKLEMIDDLE;
  Tackler5.Left := TACKLEROW3;
  Tackler5.Top := TACKLEMIDDLE;

  ShowField(true);
end;

<-------> procedure TMainForm.Sacked( Player: TLabel);
begin
  sndPlaySound('whistle.wav', SND_SYNC);
  running := false;
  LastSack := player;
  if (YardsToGo <=0) then
  begin
    Down := 1;
    YardsToGo := 10;
  end
  else
  begin
    inc( Down );
    if (Down > 4) then
    begin
      sndPlaySound('whistle.wav', SND_SYNC);
      Down := 1;  // First down for visitor
      YardsToGo := 10;
      Computer.Visible := true;
    end;
  end;
end;

<-------> procedure TMainForm.ShowDisplay(visible: boolean);
begin
  DownLabel.Visible := visible;
  FieldPosLabel.Visible := visible;
  YTGLabel.Visible := visible;
  DownDisplay.Visible := visible;
  FieldPosDisplay.Visible := visible;
  YTGDisplay.Visible := visible;
end;
```

```
    if (LastSack <> Nil) then
      LastSack.Visible := not LastSack.Visible;
    exit;
  end;

  if (field[x][y].rb) then
    exit; // can't move the runningback!

  if (field[x][y].player = Nil) then
    exit; // no tacker at this spot

  if (field[x][y].player.Left <= RBLEFT) then
    exit; // already at the endzone

  if (direction = 0)  then// we're moving horizontal
  begin
    if (x < rbx) then
      newx := x + 1
    else if (x > rbx) then
      newx := x - 1
    else
      exit;
    //  we're already horizontally lined up with rb
  end
  else if (direction = 1)  then// we're moving vertical
  begin
    if (y < rby) then
      newy := y + 1
    else if (y > rby) then
      newy := y - 1
    else
      exit;
    //  we're already vertically lined up with rb
  end;

  if field[newx][newy].rb  then// got him!
  begin
    Sacked(field[x][y].player);
    exit;
  end;
```

```
TMainForm.ClockTimer(Sender: TObject);
begin
  if not running then
    exit;

  sndPlaySound('tick.wav', SND_ASYNC);
  TimeLeft := TimeLeft - 0.1;
  TimeDisplay.Caption := FloatToStrF(TimeLeft, ffGeneral, 4, 4);

  if (TimeLeft <= 0) then
  begin
    inc( Quarter );
    TimeLeft := 15;
  end;
  if (Quarter >= 5) then
  begin
    // game over
    sndPlaySound('whistle.wav', SND_SYNC);
    sndPlaySound('whistle.wav', SND_SYNC);
    LastSack := Runningback; // hack to keep movement keys disabled
    TogglePlay(false);
    ShowField(false);
    ShowDisplay(true);
  end;
end;

<-------> procedure TMainForm.TimerTimer(Sender: TObject);
var
  x, y, newx, newy, direction : integer;
begin
  newy := random(3);
  y := newy;
  newx := random(10);
  x := newx;
  direction := random(2); // 0 is for x, 1 is for y

  if not running then
  begin
```

```
IntToStr(FieldPosition) + ' <';
  end
  else
  begin
    FieldPosDisplay.Caption := '> ' +
IntToStr(100 - FieldPosition);
  end;

  DownDisplay.Caption :=
IntToStr(Down);

  ShowField(false);
  ShowDisplay(true);
end;

<-------> procedure
TMainForm.KickClick(Sender:
TObject);
begin
  if ((running) or (Down <> 4)) then
    exit;  // button only valid just
before fourth down

  ShowField(false);
  inc(FieldPosition,random(100) );
  if (FieldPosition >= 100) then
  begin
    sndPlaySound('td.wav',
SND_SYNC);
    inc( Home, 3 );
    HomeDisplay.Caption :=
IntToStr(Home);
    FieldPosition := 80;  // Starting
position for visitor
  end
  else
  begin
    sndPlaySound('whistle.wav',
SND_SYNC);
    sndPlaySound('whistle.wav',
SND_SYNC);
  end;
  Down := 1;  // first down for visitor
  YardsToGo := 10;
  Computer.Visible := true;
  LastSack := Runningback; // hack
to keep movement keys disabled
end;

```

```

  if (field[newx][newy].player = Nil)
then// not blocked
  begin
    field[x][y].player.Left :=
field[x][y].player.Left - (22 * (x -
newx));
    field[x][y].player.Top :=
field[x][y].player.Top - (16 * (y -
newy));
    field[newx][newy].player :=
field[x][y].player;
    field[x][y].player := Nil;
  end;
end;

<-------> procedure
TMainForm.ScoreMouseUp(Sender:
TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  if ((OnOffSwitch.Position = 2) or
(running)) then
    exit;

  ShowDisplay(false);
  if not(Computer.Visible) then
    ResetField;
end;

<-------> procedure
TMainForm.ScoreMouseDown(Send
er: TObject; Button:
TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  if ((OnOffSwitch.Position = 2) or
(running)) then
    exit;

  if (Quarter = 5) then
    exit;

  YTGDisplay.Caption :=
IntToStr(YardsToGo);

  if (FieldPosition <= 50) then
  begin
    FieldPosDisplay.Caption :=
```

```
  randomize;
  ResetField;
end;

end.

\\\\\\\\\\\\\\\\\\\\
unit about;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls;

type
  TAboutForm = class(TForm)
    Label1: TLabel;
    Memo1: TMemo;
    Button1: TButton;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  AboutForm: TAboutForm;

implementation

{$R *.dfm}

end.
\\\\\\\\\\\\\\\\\
```

```
<-------> procedure
TMainForm.OnOffSwitchChange(Sender: TObject);
begin
  case OnOffSwitch.Position of
    1:
    begin
      Timer.Interval := 250;
      HomeDisplay.Visible := true;
      VisitorDisplay.Visible := true;
      TimeDisplay.Visible := true;
      ResetGame;
    end;
    2:
    begin
      ShowField(false);
      HomeDisplay.Visible := false;
      VisitorDisplay.Visible := false;
      TimeDisplay.Visible := false;
    end;
    3:
    begin
      Timer.Interval := 100;
      HomeDisplay.Visible := true;
      VisitorDisplay.Visible := true;
      TimeDisplay.Visible := true;
      ResetGame;
    end;
  end;
end;

<-------> procedure
TMainForm.ResetGame;
begin
  YardsToGo := 10;
  Down := 1;
  FieldPosition := 20;
  Home := 0;
  Visitor := 0;
  Quarter := 1;
  TimeLeft := 15;
  TimeDisplay.Caption :=
FloatToStrF(TimeLeft, ffGeneral, 4,
4);
  HomeDisplay.Caption :=
IntToStr(Home);
  VisitorDisplay.Caption :=
intToStr(Visitor);
```

القسم السابع
مجموعة من البرامج المتقدمة والشاملة
في دلفي
(متقدمين)

Objects
الملف التنفيذي لبرنامج
(AlphaBlendDemo)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

AlphaBlendDemo
49

```
program AlphaBlendDemo;
```

```
uses
  Forms,
  Main in 'Main.pas' {Form1};

{$R *.res}

begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.Run;
end.


\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

unit Main;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdActns, ExtActns, ActnList, ToolWin, ActnMan, ActnCtrls,
  ActnMenus, StdCtrls, ComCtrls, ImgList, StdActnMenus,
  XPStyleActnCtrls, XPActnCtrls;

type
  TForm1 = class(TForm)
    ActionManager1: TActionManager;
    ActionMainMenuBar1: TActionMainMenuBar;
    RichEdit1: TRichEdit;
    ImageList1: TImageList;
    EditCut1: TEditCut;
    EditCopy1: TEditCopy;
    EditPaste1: TEditPaste;
    EditSelectAll1: TEditSelectAll;
    EditUndo1: TEditUndo;
    EditDelete1: TEditDelete;
    RichEditBold1: TRichEditBold;
    RichEditItalic1: TRichEditItalic;
    RichEditUnderline1: TRichEditUnderline;
    RichEditStrikeOut1: TRichEditStrikeOut;
    RichEditBullets1: TRichEditBullets;
    RichEditAlignLeft1: TRichEditAlignLeft;
    RichEditAlignRight1: TRichEditAlignRight;
    RichEditAlignCenter1: TRichEditAlignCenter;
    FileExit1: TFileExit;
    SearchFind1: TSearchFind;
    SearchFindNext1: TSearchFindNext;
    SearchReplace1: TSearchReplace;
    SearchFindFirst1: TSearchFindFirst;
    ActionToolBar1: TActionToolBar;
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
    procedure PopupClass(Sender: TObject; var PopupClass: TCustomPopupClass);
  public
    { Public declarations }
  end;

{ TCustomLayeredActionPopupEx }

  TCustomLayeredActionPopupEx = class(TXPStylePopupMenu)
  private
    FAlphaBlend: Boolean;
    FAlphaBlendValue: Byte;
    FTransparentColor: Boolean;
    FTransparentColorValue: TColor;
    procedure SetAlphaBlend(const Value: Boolean);
    procedure SetAlphaBlendValue(const Value: Byte);
    procedure SetTransparentColor(const Value: Boolean);
```

```
procedure TCustomLayeredActionPopupEx.SetAlphaBlendValue(const Value: Byte);
begin
  if FAlphaBlendValue <> Value then
  begin
    FAlphaBlendValue := Value;
    SetLayeredAttribs;
  end;
end;

procedure TCustomLayeredActionPopupEx.SetLayeredAttribs;
const
  cUseAlpha: array [Boolean] of Integer = (0, LWA_ALPHA);
  cUseColorKey: array [Boolean] of Integer = (0, LWA_COLORKEY);
var
  AStyle: Integer;
begin
  if not (csDesigning in ComponentState) and
    (Assigned(SetLayeredWindowAttributes)) and HandleAllocated then
  begin
    AStyle := GetWindowLong(Handle, GWL_EXSTYLE);
    if FAlphaBlend then
    begin
      if (AStyle and WS_EX_LAYERED) = 0 then
        SetWindowLong(Handle, GWL_EXSTYLE, AStyle or WS_EX_LAYERED);
      SetLayeredWindowAttributes(Handle, FTransparentColorValue, FAlphaBlendValue,
        cUseAlpha[FAlphaBlend] or cUseColorKey[FTransparentColor]);
    end
```

```
    procedure SetTransparentColorValue(const Value: TColor);
  protected
    procedure CreateParams(var Params: TCreateParams); override;
    procedure CreateWindowHandle(const Params: TCreateParams); override;
    procedure InitAlphaBlending(var Params: TCreateParams);
    procedure SetLayeredAttribs;
    property AlphaBlend: Boolean read FAlphaBlend write SetAlphaBlend;
    property AlphaBlendValue: Byte read FAlphaBlendValue write SetAlphaBlendValue;
    property TransparentColor: Boolean read FTransparentColor write SetTransparentColor;
    property TransparentColorValue: TColor read FTransparentColorValue write SetTransparentColorValue;
  public
    constructor Create(AOwner: TComponent); override;
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

{ TCustomLayeredActionPopup }

procedure TCustomLayeredActionPopupEx.SetAlphaBlend(const Value: Boolean);
begin
  if FAlphaBlend <> Value then
  begin
    FAlphaBlend := Value;
    SetLayeredAttribs;
  end;
end;
```

```
utes)) then
    if FAlphaBlend or
FTransparentColor then
      Params.ExStyle :=
Params.ExStyle or
WS_EX_LAYERED;
end;

procedure
TCustomLayeredActionPopupEx.Cr
eateWindowHandle(
  const Params: TCreateParams);
begin
  inherited;
  SetLayeredAttribs;
end;

constructor
TCustomLayeredActionPopupEx.Cr
eate(AOwner: TComponent);
begin
  inherited;
  FAlphaBlend := True;
  FAlphaBlendValue := 125;
  FTransparentColorValue :=
clWhite;
end;

procedure
TCustomLayeredActionPopupEx.Cr
eateParams(
  var Params: TCreateParams);
begin
  inherited;
  InitAlphaBlending(Params);
end;

procedure
TForm1.FormCreate(Sender:
TObject);
begin

ActionMainMenuBar1.OnGetPopup
Class := PopupClass;
end;

procedure
TForm1.PopupClass(Sender:
TObject;
```

```
    else
    begin
      SetWindowLong(Handle,
GWL_EXSTYLE, AStyle and not
WS_EX_LAYERED);
      RedrawWindow(Handle, nil, 0,
RDW_ERASE or
RDW_INVALIDATE or
RDW_FRAME or
RDW_ALLCHILDREN);
    end;
  end;
end;

procedure
TCustomLayeredActionPopupEx.Se
tTransparentColorValue(const
Value: TColor);
begin
  if FTransparentColorValue <>
Value then
  begin
    FTransparentColorValue :=
Value;
    SetLayeredAttribs;
  end;
end;

procedure
TCustomLayeredActionPopupEx.Se
tTransparentColor(const Value:
Boolean);
begin
  if FTransparentColor <> Value
then
  begin
    FTransparentColor := Value;
    SetLayeredAttribs;
  end;
end;

procedure
TCustomLayeredActionPopupEx.Ini
tAlphaBlending(var Params:
TCreateParams);
begin
  if not (csDesigning in
ComponentState) and
(assigned(SetLayeredWindowAttrib
```

```
end.
\\\\\\\\\\\\\\\\\\\\\\
```

```
    ColorMap.HighlightColor = 15660791
    ColorMap.BtnSelectedColor = clBtnFace
    ColorMap.UnusedColor = 15660791
    EdgeBorders = [ebTop, ebBottom]
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'Tahoma'
    Font.Style = []
    Spacing = 3
  end
  object RichEdit1: TRichEdit
    Left = 0
    Top = 53
    Width = 413
    Height = 234
    Align = alClient
    Lines.Strings = (
      'RichEdit1')
    TabOrder = 1
  end
  object ActionToolBar1: TActionToolBar
    Left = 0
    Top = 27
    Width = 413
    Height = 26
    ActionManager = ActionManager1
    Caption = 'ActionToolBar1'
    ColorMap.HighlightColor = 15660791
    ColorMap.BtnSelectedColor = clBtnFace
    ColorMap.UnusedColor = 15660791
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
```

```
  var PopupClass: TCustomPopupClass);
begin
  PopupClass := TCustomLayeredActionPopupEx;
end;
```

Object$

الملف االنصي لبرنامج

(AlphaBlendDemo)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
AlphaBlendDemo
object Form1: TForm1
  Left = 241
  Top = 161
  Width = 421
  Height = 321
  Caption = 'ActionBand AlphaBlended Menus Demo'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  object ActionMainMenuBar1: TActionMainMenuBar
    Left = 0
    Top = 0
    Width = 413
    Height = 27
    UseSystemFont = False
    ActionManager = ActionManager1
    AnimationStyle = asNone
    Caption = 'ActionMainMenuBar1'
```

```
          ShortCut = 46
        end>
      Caption = '&Edit'
    end
    item
      Items = <
        item
          Action = SearchFind1
          ImageIndex = 14
          ShortCut = 16454
        end
        item
          Action = SearchFindNext1
          ImageIndex = 15
          ShortCut = 114
        end
        item
          Action = SearchReplace1
          ImageIndex = 16
        end
        item
          Action = SearchFindFirst1
        end>
      Caption = '&Search'
    end
    item
      Items = <
        item
          Action = RichEditBold1
          ImageIndex = 5
          ShortCut = 16450
        end
        item
          Action = RichEditItalic1
          ImageIndex = 6
          ShortCut = 16457
        end
        item
          Action = RichEditUnderline1
          ImageIndex = 7
          ShortCut = 16469
        end
        item
          Action = RichEditStrikeOut1
          ImageIndex = 8
        end
```

```
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    Spacing = 0
  end
  object ActionManager1: TActionManager
    ActionBars = <
      item
        Items = <
          item
            Items = <
              item
                Action = FileExit1
                ImageIndex = 13
              end>
            Caption = '&File'
          end
          item
            Items = <
              item
                Action = EditCut1
                ImageIndex = 0
                ShortCut = 16472
              end
              item
                Action = EditCopy1
                ImageIndex = 1
                ShortCut = 16451
              end
              item
                Action = EditPaste1
                ImageIndex = 2
                ShortCut = 16470
              end
              item
                Action = EditSelectAll1
                ShortCut = 16449
              end
              item
                Action = EditUndo1
                ImageIndex = 3
                ShortCut = 16474
              end
              item
                Action = EditDelete1
                ImageIndex = 4
```

item
Action = RichEditBold1
Caption = 'B&old'
ImageIndex = 5
ShortCut = 16450
end>
ActionBar = ActionToolBar1
end>
Images = ImageList1
Left = 135
Top = 164
StyleName = 'XP Style'
object EditCut1: TEditCut
Category = 'Edit'
Caption = 'Cu&t'
Hint = 'Cut|Cuts the selection and puts it on the Clipboard'
ImageIndex = 0
ShortCut = 16472
end
object EditCopy1: TEditCopy
Category = 'Edit'
Caption = '&Copy'
Hint = 'Copy|Copies the selection and puts it on the Clipboard'
ImageIndex = 1
ShortCut = 16451
end
object EditPaste1: TEditPaste
Category = 'Edit'
Caption = '&Paste'
Hint = 'Paste|Inserts Clipboard contents'
ImageIndex = 2
ShortCut = 16470
end
object EditSelectAll1: TEditSelectAll
Category = 'Edit'
Caption = 'Select &All'
Hint = 'Select All|Selects the entire document'
ShortCut = 16449
end
object EditUndo1: TEditUndo
Category = 'Edit'
Caption = '&Undo'
Hint = 'Undo|Reverts the last

item
Action = RichEditBullets1
Caption = 'Bu&llets'
ImageIndex = 9
end
item
Action = RichEditAlignLeft1
Caption = '&Align Left'
ImageIndex = 10
end
item
Action = RichEditAlignRight1
ImageIndex = 11
end
item
Action = RichEditAlignCenter1
ImageIndex = 12
end>
Caption = 'F&ormat'
end>
ActionBar = ActionMainMenuBar1
end
item
Items = <
item
Action = RichEditAlignCenter1
ImageIndex = 12
end
item
Action = RichEditAlignRight1
ImageIndex = 11
end
item
Action = RichEditUnderline1
ImageIndex = 7
ShortCut = 16469
end
item
Action = RichEditItalic1
ImageIndex = 6
ShortCut = 16457
end

TRichEditBullets
Category = 'Format'
AutoCheck = True
Caption = '&Bullets'
Hint = 'Bullets|Inserts a bullet on the current line'
ImageIndex = 9
end
object RichEditAlignLeft1: TRichEditAlignLeft
Category = 'Format'
AutoCheck = True
Caption = 'Align &Left'
Hint = 'Align Left|Aligns text at the left indent'
ImageIndex = 10
end
object RichEditAlignRight1: TRichEditAlignRight
Category = 'Format'
AutoCheck = True
Caption = 'Align &Right'
Hint = 'Align Right|Aligns text at the right indent'
ImageIndex = 11
end
object RichEditAlignCenter1: TRichEditAlignCenter
Category = 'Format'
AutoCheck = True
Caption = '&Center'
Hint = 'Center|Centers text between margins'
ImageIndex = 12
end
object FileExit1: TFileExit
Category = 'File'
Caption = 'E&xit'
Hint = 'Exit|Quits the application'
ImageIndex = 13
end
object SearchFind1: TSearchFind
Category = 'Search'
Caption = '&Find...'
Hint = 'Find|Finds the specified text'
ImageIndex = 14

action'
ImageIndex = 3
ShortCut = 16474
end
object EditDelete1: TEditDelete
Category = 'Edit'
Caption = '&Delete'
Hint = 'Delete|Erases the selection'
ImageIndex = 4
ShortCut = 46
end
object RichEditBold1: TRichEditBold
Category = 'Format'
AutoCheck = True
Caption = '&Bold'
Hint = 'Bold'
ImageIndex = 5
ShortCut = 16450
end
object RichEditItalic1: TRichEditItalic
Category = 'Format'
AutoCheck = True
Caption = '&Italic'
Hint = 'Italic'
ImageIndex = 6
ShortCut = 16457
end
object RichEditUnderline1: TRichEditUnderline
Category = 'Format'
AutoCheck = True
Caption = '&Underline'
Hint = 'Underline'
ImageIndex = 7
ShortCut = 16469
end
object RichEditStrikeOut1: TRichEditStrikeOut
Category = 'Format'
AutoCheck = True
Caption = '&Strikeout'
Hint = 'Strikeout'
ImageIndex = 8
end
object RichEditBullets1:

```
uses
  Forms,
  main in 'main.pas' {Form1};

{$R *.res}

begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.Run;
end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
unit main;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ActnMan, ActnList, ToolWin, ActnCtrls, ActnMenus, StdCtrls,
  StdActns, ExtCtrls, ComCtrls, XPStyleActnCtrls;

type
  TForm1 = class(TForm)
    ActionManager1: TActionManager;
    Action1: TAction;
    Action2: TAction;
    ActionMainMenuBar1: TActionMainMenuBar;
    EditCut1: TEditCut;
    EditCopy1: TEditCopy;
    EditPaste1: TEditPaste;
    EditSelectAll1: TEditSelectAll;
    EditUndo1: TEditUndo;
    Memo1: TMemo;
    Panel1: TPanel;
    Button4: TButton;
    Button1: TButton;
    Button3: TButton;
    Button2: TButton;
    AddCategoryAction: TAction;
```

```
      ShortCut = 16454
    end
    object SearchFindNext1: TSearchFindNext
      Category = 'Search'
      Caption = 'Find &Next'
      Hint = 'Find Next|Repeats the last find'
      ImageIndex = 15
      ShortCut = 114
    end
    object SearchReplace1: TSearchReplace
      Category = 'Search'
      Caption = '&Replace'
      Hint = 'Replace|Replaces specific text with different text'
      ImageIndex = 16
    end
    object SearchFindFirst1: TSearchFindFirst
      Category = 'Search'
      Caption = 'F&ind First'
      Hint = 'Find First|Finds the first occurance of specified text'
    end
  end
  object ImageList1: TImageList
    Left = 114
    Top = 76
    Bitmap = {
      }
  end
end
```

ObjectS
الملف التنفيذي لبرنامج
(DynaActionBands)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
DynaActionBands
program DynaActionBands;
```

```
    AddActionAction: TAction;
    AddSeparatorAction: TAction;
    DeleteItemsAction: TAction;
    StatusBar1: TStatusBar;
    DisableBtn: TButton;
    ActionList1: TActionList;
    EditDelete2: TEditDelete;
    procedure DeleteItemsActionExecute(Sender: TObject);
    procedure AddSeparatorActionExecute(Sender: TObject);
    procedure AddCategoryActionExecute(Sender: TObject);
    procedure AddActionActionExecute(Sender: TObject);
    procedure DisableBtnClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.DeleteItemsActionExecute(Sender: TObject);
begin
  ActionManager1.DeleteActionItems([Action1]);
end;

procedure TForm1.AddSeparatorActionExecute(Sender: TObject);
var
  Item: TActionClientItem;
begin
```

```
  Item := ActionManager1.FindItemByAction(Action1);
  if Assigned(Item) then
    ActionManager1.AddSeparator(Item);
end;

procedure TForm1.AddCategoryActionExecute(Sender: TObject);
var
  Item: TActionClientItem;
begin
  Item := ActionManager1.FindItemByCaption('&test');
  if Assigned(Item) then
    ActionManager1.AddCategory('Edit', Item);
end;

procedure TForm1.AddActionActionExecute(Sender: TObject);
var
  Item: TActionClientItem;
begin
  Item := ActionManager1.FindItemByCaption('&Action1');
  if Assigned(Item) then
    ActionManager1.AddAction(EditCut1, Item);
end;

procedure TForm1.DisableBtnClick(Sender: TObject);
var
  Item: TActionClientItem;
begin
  Item := ActionManager1.FindItemByCaption('&test');
  if Assigned(Item) then
```

```
    Item.Control.Enabled := not
Item.Control.Enabled;
end;
```

Object$
الملف النصي لبرنامج
(DynaActionBands)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

```
DynaActionBands
object Form1: TForm1
  Left = -63
  Top = 117
  Width = 498
  Height = 338
  Caption = 'ActionManager Helper
Demo'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  ShowHint = True
  PixelsPerInch = 96
  TextHeight = 13
  object ActionMainMenuBar1:
TActionMainMenuBar
    Left = 0
    Top = 0
    Width = 490
    Height = 24
    UseSystemFont = False
    ActionManager =
ActionManager1
    Caption = 'ActionMainMenuBar1'
    ColorMap.HighlightColor =
15660791
    ColorMap.BtnSelectedColor =
clBtnFace
    ColorMap.UnusedColor =
```

```

end.
```

```
15660791
    Font.Charset = ANSI_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'Tahoma'
    Font.Style = []
    Spacing = 3
  end
  object Memo1: TMemo
    Left = 0
    Top = 60
    Width = 490
    Height = 225
    Align = alClient
    TabOrder = 1
  end
  object Panel1: TPanel
    Left = 0
    Top = 24
    Width = 490
    Height = 36
    Align = alTop
    BevelOuter = bvNone
    TabOrder = 2
    object Button4: TButton
      Left = 260
      Top = 6
      Width = 102
      Height = 25
      Action = DeleteItemsAction
      TabOrder = 3
    end
    object Button1: TButton
      Left = 171
      Top = 6
      Width = 82
      Height = 25
      Action = AddSeparatorAction
      TabOrder = 2
    end
    object Button3: TButton
      Left = 8
      Top = 6
      Width = 75
      Height = 25
```

```
    end>
  LinkedActionLists = <
    item
      ActionList = ActionList1
      Caption = 'ActionList1'
    end>
  Left = 61
  Top = 73
  StyleName = 'XP Style'
  object Action1: TAction
    Category = 'test'
    Caption = 'Action1'
    Enabled = False
  end
  object Action2: TAction
    Category = 'Test'
    Caption = 'Action2'
    Enabled = False
  end
  object EditCut1: TEditCut
    Category = 'Edit'
    Caption = 'Cu&t'
    Enabled = False
    Hint = 'Cut|Cuts the selection
and puts it on the Clipboard'
    ImageIndex = 0
    ShortCut = 16472
  end
  object EditCopy1: TEditCopy
    Category = 'Edit'
    Caption = '&Copy'
    Enabled = False
    Hint = 'Copy|Copies the selection
and puts it on the Clipboard'
    ImageIndex = 1
    ShortCut = 16451
  end
  object EditPaste1: TEditPaste
    Category = 'Edit'
    Caption = '&Paste'
    Enabled = False
    Hint = 'Paste|Inserts Clipboard
contents'
    ImageIndex = 2
    ShortCut = 16470
  end
  object EditSelectAll1:
TEditSelectAll
```

```
    Action = AddCategoryAction
    TabOrder = 0
  end
  object Button2: TButton
    Left = 89
    Top = 6
    Width = 75
    Height = 25
    Action = AddActionAction
    TabOrder = 1
  end
  object DisableBtn: TButton
    Left = 372
    Top = 6
    Width = 112
    Height = 25
    Caption = 'Disable Menu Button'
    TabOrder = 4
    OnClick = DisableBtnClick
  end
 end
 object StatusBar1: TStatusBar
  Left = 0
  Top = 285
  Width = 490
  Height = 19
  AutoHint = True
  Panels = <>
 end
 object ActionManager1:
TActionManager
  ActionBars = <
    item
      Items = <
        item
          Items = <
            item
              Action = Action1
              Caption = '&Action1'
            end
            item
              Action = Action2
              Caption = 'A&ction2'
            end>
          Caption = '&test'
        end>
      ActionBar =
ActionMainMenuBar1
```

```
    object DeleteItemsAction: TAction
      Category = 'Buttons'
      Caption = 'Delete Action Items'
      Hint = 'Deletes all items attached to Action1'
      OnExecute = DeleteItemsActionExecute
    end
  end
  object ActionList1: TActionList
    Left = 194
    Top = 74
    object EditDelete2: TEditDelete
      Category = 'Edit'
      Caption = '&Delete'
      Hint = 'Delete|Erases the selection'
      ImageIndex = 5
      ShortCut = 46
    end
  end
end
```

Object$
الملف التنفيذي لبرنامج
(MRU)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

MRU

```
      Category = 'Edit'
      Caption = 'Select &All'
      Enabled = False
      Hint = 'Select All|Selects the entire document'
      ShortCut = 16449
    end
    object EditUndo1: TEditUndo
      Category = 'Edit'
      Caption = '&Undo'
      Enabled = False
      Hint = 'Undo|Reverts the last action'
      ImageIndex = 3
      ShortCut = 16474
    end
    object AddCategoryAction: TAction
      Category = 'Buttons'
      Caption = 'Add Category'
      Hint =
        'Adds the "Edit" Category to the menu after the item with the cap' +
        'tion "&Test"'
      OnExecute = AddCategoryActionExecute
    end
    object AddActionAction: TAction
      Category = 'Buttons'
      Caption = 'Add Action'
      Hint =
        'Adds the EditCut action after the item with the caption "&Action' +
        '1"'
      OnExecute = AddActionActionExecute
    end
    object AddSeparatorAction: TAction
      Category = 'Buttons'
      Caption = 'Add Separator'
      Hint = 'Adds a separator after the item that is connected to Action1'
      OnExecute = AddSeparatorActionExecute
    end
```

```
program ABMRU;

uses
  Forms,
  main in 'main.pas' {Form1};

{$R *.res}

begin
  Application.Initialize;
  Application.CreateForm(TForm1,
Form1);
  Application.Run;
end.

{
   NOTE: To add more items to the
reopen menu simply add more
actions to the
         ReopenActionList and set their
OnExecute event to the
ReopenActionExecute
         event.

         Also, in order for the settings
file to be saved you must run and
         close this application at least
one time.  So, to see the reopen
         items working run the
application and shut it down first
then use
         it normally.

   ISSUE: If you allow
customizations in your application
there is potential
          for the user to delete the
Reopen menu item or the Open
toolbar button
          which this application does
NOT take into account.
}
unit main;

interface

uses
  Windows, Messages, SysUtils,
Variants, Classes, Graphics,
Controls, Forms,
  Dialogs, ActnList, StdActns,
ToolWin, ActnMan, ActnCtrls,
ActnMenus,
  StdCtrls, ImgList, ComCtrls,
ExtActns, XPStyleActnCtrls,
BandActn,
  StdStyleActnCtrls;

type
  TForm1 = class(TForm)
    ActionManager1:
TActionManager;
    ActionMainMenuBar1:
TActionMainMenuBar;
    Action1: TAction;
    Action2: TAction;
    Action3: TAction;
    ReopenActionList1: TActionList;
    Action4: TAction;
    Action5: TAction;
    Action6: TAction;
    Action7: TAction;
    Action8: TAction;
    ActionToolBar1:
TActionToolBar;
    ImageList1: TImageList;
    FileOpen1: TFileOpen;
    EditCut1: TEditCut;
    EditCopy1: TEditCopy;
    EditPaste1: TEditPaste;
    EditSelectAll1: TEditSelectAll;
    EditUndo1: TEditUndo;
    EditDelete1: TEditDelete;
    RichEditBold1: TRichEditBold;
    RichEditItalic1: TRichEditItalic;
    RichEditUnderline1:
TRichEditUnderline;
    RichEditStrikeOut1:
TRichEditStrikeOut;
    RichEditBullets1:
TRichEditBullets;
    RichEditAlignLeft1:
TRichEditAlignLeft;
    RichEditAlignRight1:
TRichEditAlignRight;
    RichEditAlignCenter1:
TRichEditAlignCenter;
    SearchFind1: TSearchFind;
```

```
    SearchFindNext1: TSearchFindNext;
    SearchReplace1: TSearchReplace;
    SearchFindFirst1: TSearchFindFirst;
    FileSaveAs1: TFileSaveAs;
    FilePrintSetup1: TFilePrintSetup;
    FileRun1: TFileRun;
    FileExit1: TFileExit;
    Action9: TAction;
    RichEdit1: TRichEdit;
    CustomizeActionBars1: TCustomizeActionBars;
    procedure FormCreate(Sender: TObject);
    procedure FileOpen1Accept(Sender: TObject);
    procedure ReopenActionExecute(Sender: TObject);
  private
    { Private declarations }
    ReopenMenuItem: TActionClientItem;
    OpenToolItem: TActionClientItem;
    procedure FindReopenMenuItem(AClient: TActionClient);
    procedure FindOpenToolItem(AClient: TActionClient);
    procedure UpdateReopenItem(ReopenItem: TActionClientItem);
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.FindReopenMenuItem(AClient: TActionClient);
begin
  // Find the Reopen item by looking at the item caption
  if AClient is TActionClientItem then
    if Pos('Reopen...', TActionClientItem(AClient).Caption) <> 0 then
      ReopenMenuItem := AClient as TActionClientItem
end;

procedure TForm1.FindOpenToolItem(AClient: TActionClient);
begin
  // Find the Open item by looking at the item caption
  if AClient is TActionClientItem then
    if Pos('Open', TActionClientItem(AClient).Caption) <> 0 then
      OpenToolItem := AClient as TActionClientItem;
end;

procedure TForm1.FormCreate(Sender: TObject);

  procedure SetupItemCaptions(AnItem: TActionClientItem);
  var
    I: Integer;
  begin
    if Assigned(AnItem) then
      for I := 0 to AnItem.Items.Count - 1 do
        TCustomAction(ReopenActionList1.Actions[I]).Caption :=
          Copy(AnItem.Items[I].Caption, 5, MaxInt);
  end;

begin
```

```
UpdateReopenItem(ReopenMenuItem);
UpdateReopenItem(OpenToolItem);
  end;
  // ...then actually open the file
RichEdit1.Lines.LoadFromFile(FileOpen1.Dialog.FileName);
end;

procedure TForm1.UpdateReopenItem(ReopenItem: TActionClientItem);
var
  I: Integer;
begin
  if ReopenItem = nil then exit;
  // Add thew new filename to the beginning of the list and move other items down
  for I := ReopenActionList1.ActionCount - 1 downto 0 do
    if I = 0 then
TCustomAction(ReopenActionList1.Actions[I]).Caption := FileOpen1.Dialog.FileName
    else
TCustomAction(ReopenActionList1.Actions[I]).Caption :=
TCustomAction(ReopenActionList1.Actions[I - 1]).Caption;
  // Add new items to the reopen item if necessary
  if ReopenItem.Items.Count < ReopenActionList1.ActionCount then
    ReopenItem.Items.Add;
  // Set the item captions by appending a number for use as the shortcut
  // This change will cause them to be streamed which allows us to store the
  RichEdit1.Align := alClient;
  // Find the Reopen... menu item on the ActionMainMenu
ActionManager1.ActionBars.IterateClients(ActionManager1.ActionBars[0].Items,
    FindReopenMenuItem);
  // Find the Reopen... menu item on the ActionToolBar
ActionManager1.ActionBars.IterateClients(ActionManager1.ActionBars[1].Items,
    FindOpenToolItem);
  // Set the captions of the actions since they are used to open the file
SetupItemCaptions(ReopenMenuItem);
SetupItemCaptions(OpenToolItem);
end;

procedure TForm1.FileOpen1Accept(Sender: TObject);
var
  I: Integer;
  Found: Boolean;
begin
  Found := False;
  // If the filename is already in the list then do not add it again
  for I := 0 to ReopenActionList1.ActionCount - 1 do
    if CompareText(TCustomAction(ReopenActionList1.Actions[I]).Caption,
      FileOpen1.Dialog.FileName) = 0 then
    begin
      Found := True;
      break;
    end;
  if not Found then
  begin
    // Update the Reopen menu...
```

```
  Width = 499
  Height = 285
  Caption = 'ActionBands Reopen Demo'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  object ActionMainMenuBar1: TActionMainMenuBar
    Left = 0
    Top = 0
    Width = 491
    Height = 25
    UseSystemFont = False
    ActionManager = ActionManager1
    Caption = 'ActionMainMenuBar1'
    ColorMap.HighlightColor = 15660791
    ColorMap.BtnSelectedColor = clBtnFace
    ColorMap.UnusedColor = 15660791
    EdgeBorders = [ebBottom]
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'Tahoma'
    Font.Style = []
    Spacing = 3
  end
  object ActionToolBar1: TActionToolBar
    Left = 0
    Top = 25
    Width = 491
    Height = 26
    ActionManager = ActionManager1
```

```
  // filenames when the application is shutdown
  for I := 0 to ReopenItem.Items.Count - 1 do
  begin
    ReopenItem.Items[I].Action := ReopenActionList1.Actions[I];
    ReopenItem.Items[I].Caption := Format('&%d: %s', [I,

TCustomAction(ReopenActionList1.Actions[I]).Caption]);
  end;
end;

procedure TForm1.ReopenActionExecute(Sender: TObject);
begin
  // Set the reopened filename into the FileOpen action and call OnAccept to open the file normally
  FileOpen1.Dialog.FileName := (Sender as TCustomAction).Caption;
  // Execute the action's OnAccept logic
  FileOpen1.OnAccept(nil);
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\
```

Object$

الملف النصي لبرنامج

(MRU)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
MRU
object Form1: TForm1
  Left = 187
  Top = 200
```

```
    end
    object Action8: TAction
      Caption = 'Action8'
      OnExecute = ReopenActionExecute
    end
  end
  object ActionManager1: TActionManager
    FileName = 'c:\code\actions\action bars\reopen\settings.dat'
    ActionBars = <
      item
        Items = <
          item
            ChangesAllowed = [caModify]
            Items.HideUnused = False
            Items = <
              item
                Action = FileOpen1
                ImageIndex = 0
                ShortCut = 16463
              end
              item
                ChangesAllowed = [caModify]
                Caption = '&Reopen...'
              end
              item
                Caption = '-'
              end
              item
                Action = FileExit1
                ImageIndex = 19
              end>
            Caption = '&File'
          end
          item
            Items = <
              item
                Action = EditCut1
                ImageIndex = 1
                ShortCut = 16472
              end
              item
                Action = EditCopy1
                ImageIndex = 2
```

```
    Caption = 'ActionToolBar1'
    ColorMap.HighlightColor = 15660791
    ColorMap.BtnSelectedColor = clBtnFace
    ColorMap.UnusedColor = 15660791
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    Spacing = 0
  end
  object RichEdit1: TRichEdit
    Left = 9
    Top = 65
    Width = 371
    Height = 182
    Lines.Strings = (
      'RichEdit1')
    TabOrder = 2
  end
  object ReopenActionList1: TActionList
    Left = 91
    Top = 94
    object Action4: TAction
      Caption = 'Action4'
      OnExecute = ReopenActionExecute
    end
    object Action5: TAction
      Caption = 'Action5'
      OnExecute = ReopenActionExecute
    end
    object Action6: TAction
      Caption = 'Action6'
      OnExecute = ReopenActionExecute
    end
    object Action7: TAction
      Caption = 'Action7'
      OnExecute = ReopenActionExecute
```

```
          ImageIndex = 6
          ShortCut = 16450
        end
        item
          Action = RichEditItalic1
          ImageIndex = 7
          ShortCut = 16457
        end
        item
          Action =
RichEditUnderline1
          ImageIndex = 8
          ShortCut = 16469
        end
        item
          Action =
RichEditStrikeOut1
          ImageIndex = 9
        end
        item
          Action = RichEditBullets1
          Caption = 'Bu&llets'
          ImageIndex = 10
        end
        item
          Action =
RichEditAlignLeft1
          Caption = '&Align Left'
          ImageIndex = 11
        end
        item
          Action =
RichEditAlignRight1
          ImageIndex = 12
        end
        item
          Action =
RichEditAlignCenter1
          ImageIndex = 13
        end>
      Caption = 'F&ormat'
    end
    item
      Items = <
        item
          Action =
CustomizeActionBars1
        end>
```

```
          ShortCut = 16451
        end
        item
          Action = EditPaste1
          ImageIndex = 3
          ShortCut = 16470
        end
        item
          Action = EditSelectAll1
          ShortCut = 16449
        end
        item
          Action = EditUndo1
          ImageIndex = 4
          ShortCut = 16474
        end
        item
          Action = EditDelete1
          ImageIndex = 5
          ShortCut = 46
        end>
      Caption = '&Edit'
    end
    item
      Items = <
        item
          Action = SearchFind1
          ImageIndex = 14
          ShortCut = 16454
        end
        item
          Action = SearchFindNext1
          ImageIndex = 15
          ShortCut = 114
        end
        item
          Action = SearchReplace1
          ImageIndex = 16
        end
        item
          Action = SearchFindFirst1
        end>
      Caption = '&Search'
    end
    item
      Items = <
        item
          Action = RichEditBold1
```

```
    ShortCut = 16463
    OnAccept = FileOpen1Accept
  end
  object EditCut1: TEditCut
    Category = 'Edit'
    Caption = 'Cu&t'
    Hint = 'Cut|Cuts the selection and puts it on the Clipboard'
    ImageIndex = 1
    ShortCut = 16472
  end
  object EditCopy1: TEditCopy
    Category = 'Edit'
    Caption = '&Copy'
    Hint = 'Copy|Copies the selection and puts it on the Clipboard'
    ImageIndex = 2
    ShortCut = 16451
  end
  object EditPaste1: TEditPaste
    Category = 'Edit'
    Caption = '&Paste'
    Hint = 'Paste|Inserts Clipboard contents'
    ImageIndex = 3
    ShortCut = 16470
  end
  object EditSelectAll1: TEditSelectAll
    Category = 'Edit'
    Caption = 'Select &All'
    Hint = 'Select All|Selects the entire document'
    ShortCut = 16449
  end
  object EditUndo1: TEditUndo
    Category = 'Edit'
    Caption = '&Undo'
    Hint = 'Undo|Reverts the last action'
    ImageIndex = 4
    ShortCut = 16474
  end
  object EditDelete1: TEditDelete
    Category = 'Edit'
    Caption = '&Delete'
    Hint = 'Delete|Erases the selection'
```

```
          Caption = '&Tools'
        end>
      ActionBar = ActionMainMenuBar1
    end
    item
      Items.HideUnused = False
      Items = <
        item
          Action = FileOpen1
          ImageIndex = 0
          ShortCut = 16463
        end>
      ActionBar = ActionToolBar1
    end>
  LinkedActionLists = <
    item
      ActionList = ReopenActionList1
      Caption = 'Reopenlist'
    end>
  Images = ImageList1
  Left = 259
  Top = 101
  StyleName = 'XP Style'
  object Action1: TAction
    Category = 'Reopen'
    Caption = 'Action1'
    OnExecute = ReopenActionExecute
  end
  object Action2: TAction
    Category = 'Reopen'
    Caption = 'Action2'
    OnExecute = ReopenActionExecute
  end
  object Action3: TAction
    Category = 'Reopen'
    Caption = 'Action3'
    OnExecute = ReopenActionExecute
  end
  object FileOpen1: TFileOpen
    Category = 'File'
    Caption = '&Open...'
    Hint = 'Open|Opens an existing file'
    ImageIndex = 0
```

```
TRichEditAlignLeft
    Category = 'Format'
    AutoCheck = True
    Caption = 'Align &Left'
    Hint = 'Align Left|Aligns text at the left indent'
    ImageIndex = 11
  end
  object RichEditAlignRight1: TRichEditAlignRight
    Category = 'Format'
    AutoCheck = True
    Caption = 'Align &Right'
    Hint = 'Align Right|Aligns text at the right indent'
    ImageIndex = 12
  end
  object RichEditAlignCenter1: TRichEditAlignCenter
    Category = 'Format'
    AutoCheck = True
    Caption = '&Center'
    Hint = 'Center|Centers text between margins'
    ImageIndex = 13
  end
  object SearchFind1: TSearchFind
    Category = 'Search'
    Caption = '&Find...'
    Hint = 'Find|Finds the specified text'
    ImageIndex = 14
    ShortCut = 16454
  end
  object SearchFindNext1: TSearchFindNext
    Category = 'Search'
    Caption = 'Find &Next'
    Hint = 'Find Next|Repeats the last find'
    ImageIndex = 15
    SearchFind = SearchFind1
    ShortCut = 114
  end
  object SearchReplace1: TSearchReplace
    Category = 'Search'
    Caption = '&Replace'
    ImageIndex = 5
    ShortCut = 46
  end
  object RichEditBold1: TRichEditBold
    Category = 'Format'
    AutoCheck = True
    Caption = '&Bold'
    Hint = 'Bold'
    ImageIndex = 6
    ShortCut = 16450
  end
  object RichEditItalic1: TRichEditItalic
    Category = 'Format'
    AutoCheck = True
    Caption = '&Italic'
    Hint = 'Italic'
    ImageIndex = 7
    ShortCut = 16457
  end
  object RichEditUnderline1: TRichEditUnderline
    Category = 'Format'
    AutoCheck = True
    Caption = '&Underline'
    Hint = 'Underline'
    ImageIndex = 8
    ShortCut = 16469
  end
  object RichEditStrikeOut1: TRichEditStrikeOut
    Category = 'Format'
    AutoCheck = True
    Caption = '&Strikeout'
    Hint = 'Strikeout'
    ImageIndex = 9
  end
  object RichEditBullets1: TRichEditBullets
    Category = 'Format'
    AutoCheck = True
    Caption = '&Bullets'
    Hint = 'Bullets|Inserts a bullet on the current line'
    ImageIndex = 10
  end
  object RichEditAlignLeft1:
```

```
    end
    object FileExit1: TFileExit
      Category = 'File'
      Caption = 'E&xit'
      Hint = 'Exit|Quits the application'
      ImageIndex = 19
    end
    object Action9: TAction
      Category = 'File'
      Caption = 'Action9'
    end
    object CustomizeActionBars1: TCustomizeActionBars
      Category = 'Tools'
      Caption = '&Customize'
      CustomizeDlg.StayOnTop = False
    end
  end
  object ImageList1: TImageList
    Left = 178
    Top = 147
    Bitmap = {
      494C010114001900040010001000FFFFFFFFFF10FFFFFFFFFFFFFFFF424D3600
  end
end
```

```
      Hint = 'Replace|Replaces specific text with different text'
      ImageIndex = 16
    end
    object SearchFindFirst1: TSearchFindFirst
      Category = 'Search'
      Caption = 'F&ind First'
      Hint = 'Find First|Finds the first occurance of specified text'
    end
    object FileSaveAs1: TFileSaveAs
      Category = 'File'
      Caption = 'Save &As...'
      Hint = 'Save As|Saves the active file with a new name'
      ImageIndex = 18
    end
    object FilePrintSetup1: TFilePrintSetup
      Category = 'File'
      Caption = 'Print Set&up...'
      Hint = 'Print Setup'
    end
    object FileRun1: TFileRun
      Category = 'File'
      Browse = False
      BrowseDlg.Title = 'Run'
      Caption = '&Run...'
      Hint = 'Run|Runs an application'
      Operation = 'open'
      ShowCmd = scShowNormal
```

ObjectS
الملف النصي لبرنامج
(ActionBands)
إعداد

علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١
ActionBands
object Form1: TForm1
Left = 202
Top = 114
Width = 565
Height = 368
Caption = 'ActionBars Demo'
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = False
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
object ActionMainMenuBar1: TActionMainMenuBar
Left = 0
Top = 0
Width = 557
Height = 24
UseSystemFont = False
ActionManager = ActionManager1
Caption = 'ActionMainMenuBar1'
ColorMap.HighlightColor = clBtnHighlight
ColorMap.UnusedColor = 15988985
ColorMap.SelectedColor = clHighlight
EdgeBorders = [ebBottom]
EdgeOuter = esNone
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'Tahoma'
Font.Style = []
Spacing = 4
end

object ToolActionBar1: TActionToolBar
Left = 0
Top = 24
Width = 557
Height = 28
ActionManager = ActionManager1
Caption = 'ToolActionBar1'
ColorMap.HighlightColor = clBtnHighlight
ColorMap.UnusedColor = 15988985
ColorMap.SelectedColor = clHighlight
EdgeBorders = [ebLeft, ebTop, ebRight, ebBottom]
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
Spacing = 0
end
object ToolActionBar2: TActionToolBar
Left = 0
Top = 52
Width = 557
Height = 28
ActionManager = ActionManager1
Caption = 'ToolActionBar2'
ColorMap.HighlightColor = clBtnHighlight
ColorMap.UnusedColor = 15988985
ColorMap.SelectedColor = clHighlight
EdgeBorders = [ebLeft, ebTop, ebRight, ebBottom]
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'

```
  object ActionManager1: TActionManager
    ActionBars.SessionCount = 6
    ActionBars = <
      item
        Items = <
          item
            Items = <
              item
                Action = FileOpen1
                ImageIndex = 30
                ShortCut = 16463
              end
              item
                Action = FileSaveAs1
                ImageIndex = 8
              end
              item
                Action = FileRun1
                UsageCount = -1
              end
              item
                Caption = '-'
              end
              item
                Action = FileExit1
                ImageIndex = 43
                UsageCount = -1
              end>
            Caption = '&File'
          end
          item
            Items = <
              item
                Action = EditCut1
                ImageIndex = 0
                ShortCut = 16472
              end
              item
                Action = EditCopy1
                ImageIndex = 1
                ShortCut = 16451
              end
              item
                Action = EditPaste1
                ImageIndex = 2
                ShortCut = 16470
              end
```

```
    Font.Style = []
    ParentFont = False
    Spacing = 0
  end
  object ToolActionBar3: TActionToolBar
    Left = 0
    Top = 80
    Width = 557
    Height = 28
    ActionManager = ActionManager1
    Caption = 'ToolActionBar3'
    ColorMap.HighlightColor = clBtnHighlight
    ColorMap.UnusedColor = 15988985
    ColorMap.SelectedColor = clHighlight
    EdgeBorders = [ebLeft, ebTop, ebRight, ebBottom]
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    Spacing = 0
  end
  object RichEdit1: TRichEdit
    Left = 0
    Top = 108
    Width = 557
    Height = 207
    Align = alClient
    BevelKind = bkTile
    BorderStyle = bsNone
    TabOrder = 4
  end
  object StatusBar1: TStatusBar
    Left = 0
    Top = 315
    Width = 557
    Height = 19
    AutoHint = True
    Panels = <>
  end
```

end
item
Action = RichEditUnderline1
ImageIndex = 28
ShortCut = 16469
end
item
Action = RichEditStrikeOut1
end
item
Action = RichEditBullets1
Caption = 'Bull&ets'
ImageIndex = 38
end
item
Action = RichEditAlignLeft1
ImageIndex = 35
end
item
Action = RichEditAlignRight1
ImageIndex = 36
end
item
Action = RichEditAlignCenter1
ImageIndex = 37
end>
Caption = 'F&ormat'
end
item
Items = <
item
Action = CustomizeActionBars1
UsageCount = -1
end>
Caption = '&Tools'
end
item
Items = <
item
Action = StdStyleActn
Caption = '&Standard Style'

item
Action = EditSelectAll1
ShortCut = 16449
end
item
Action = EditUndo1
ImageIndex = 3
ShortCut = 16474
end
item
Action = EditDelete1
ImageIndex = 5
ShortCut = 46
end>
Caption = '&Edit'
end
item
Items = <
item
Action = SearchFind1
ImageIndex = 34
ShortCut = 16454
end
item
Action = SearchFindNext1
ImageIndex = 33
ShortCut = 114
end
item
Action = SearchReplace1
ImageIndex = 32
end
item
Action = SearchFindFirst1
end>
Caption = '&Search'
end
item
Items = <
item
Action = RichEditBold1
ImageIndex = 31
ShortCut = 16450
end
item
Action = RichEditItalic1
ImageIndex = 29
ShortCut = 16457

```
        ImageIndex = 35
      end
      item
        Action = RichEditAlignRight1
        ImageIndex = 36
      end
      item
        Action = RichEditAlignCenter1
        ImageIndex = 37
      end>
    ActionBar = ToolActionBar1
  end
  item
    Items = <
      item
        Action = SearchFind1
        ImageIndex = 34
        ShortCut = 16454
      end
      item
        Action = SearchFindNext1
        ImageIndex = 33
        ShortCut = 114
      end
      item
        Action = SearchReplace1
        ImageIndex = 32
      end
      item
        Caption = '-'
        LastSession = 6
      end
      item
        Action = SearchFindFirst1
      end>
    ActionBar = ToolActionBar2
    AutoSize = False
  end
  item
    Items = <
      item
        Action = EditCut1
        ImageIndex = 0
        ShortCut = 16472
      end
      item
```

```
            LastSession = 6
          end
          item
            Action = XPStyleActn
            Caption = '&XP Style'
            LastSession = 6
          end
          item
            Caption = '-'
          end
          item
            Action = ShadowActn
            Caption = '&Menu Shadows'
            LastSession = 6
          end>
        Caption = 'Sty&le'
      end>
    ActionBar = ActionMainMenuBar1
  end
  item
    Items = <
      item
        Action = RichEditBold1
        ImageIndex = 31
        ShortCut = 16450
      end
      item
        Action = RichEditItalic1
        ImageIndex = 29
        ShortCut = 16457
      end
      item
        Action = RichEditUnderline1
        ImageIndex = 28
        ShortCut = 16469
      end
      item
        Action = RichEditStrikeOut1
      end
      item
        Action = RichEditBullets1
        Caption = 'Bull&ets'
        ImageIndex = 38
      end
      item
        Action = RichEditAlignLeft1
```

ShortCut = 16451
end
object EditPaste1: TEditPaste
Category = 'Edit'
Caption = '&Paste'
Enabled = False
Hint = 'Paste|Inserts Clipboard contents'
ImageIndex = 2
ShortCut = 16470
end
object EditSelectAll1: TEditSelectAll
Category = 'Edit'
Caption = 'Select &All'
Enabled = False
Hint = 'Select All|Selects the entire document'
ShortCut = 16449
end
object EditUndo1: TEditUndo
Category = 'Edit'
Caption = '&Undo'
Enabled = False
Hint = 'Undo|Reverts the last action'
ImageIndex = 3
ShortCut = 16474
end
object EditDelete1: TEditDelete
Category = 'Edit'
Caption = '&Delete'
Enabled = False
Hint = 'Delete|Erases the selection'
ImageIndex = 5
ShortCut = 46
end
object RichEditBold1: TRichEditBold
Category = 'Format'
AutoCheck = True
Caption = '&Bold'
Enabled = False
Hint = 'Bold'
ImageIndex = 31
ShortCut = 16450
end

Action = EditCopy1
ImageIndex = 1
ShortCut = 16451
end
item
Action = EditPaste1
ImageIndex = 2
ShortCut = 16470
end
item
Action = EditSelectAll1
ShortCut = 16449
end
item
Action = EditUndo1
ImageIndex = 3
ShortCut = 16474
end
item
Action = EditDelete1
ImageIndex = 5
ShortCut = 46
end>
ActionBar = ToolActionBar3
AutoSize = False
end>
Images = ImageList1
OnStateChange = ActionManager1StateChange
Left = 125
Top = 138
StyleName = 'Standard'
object EditCut1: TEditCut
Category = 'Edit'
Caption = 'Cu&t'
Enabled = False
Hint = 'Cut|Cuts the selection and puts it on the Clipboard'
ImageIndex = 0
ShortCut = 16472
end
object EditCopy1: TEditCopy
Category = 'Edit'
Caption = '&Copy'
Enabled = False
Hint = 'Copy|Copies the selection and puts it on the Clipboard'
ImageIndex = 1

```
  object RichEditAlignRight1: TRichEditAlignRight
    Category = 'Format'
    AutoCheck = True
    Caption = 'Align &Right'
    Enabled = False
    Hint = 'Align Right|Aligns text at the right indent'
    ImageIndex = 36
  end
  object RichEditAlignCenter1: TRichEditAlignCenter
    Category = 'Format'
    AutoCheck = True
    Caption = '&Center'
    Enabled = False
    Hint = 'Center|Centers text between margins'
    ImageIndex = 37
  end
  object FileOpen1: TFileOpen
    Category = 'File'
    Caption = '&Open...'
    Hint = 'Open|Opens an existing file'
    ImageIndex = 30
    ShortCut = 16463
    OnAccept = FileOpen1Accept
  end
  object FileSaveAs1: TFileSaveAs
    Category = 'File'
    Caption = 'Save &As...'
    Hint = 'Save As|Saves the active file with a new name'
    ImageIndex = 8
    OnAccept = FileSaveAs1Accept
  end
  object FileRun1: TFileRun
    Category = 'File'
    Browse = False
    BrowseDlg.Title = 'Run'
    Caption = '&Run...'
    Hint = 'Run|Runs an applicatoin'
    Operation = 'open'
    ShowCmd = scShowNormal
  end
  object FileExit1: TFileExit
    Category = 'File'
```

```
  object RichEditItalic1: TRichEditItalic
    Category = 'Format'
    AutoCheck = True
    Caption = '&Italic'
    Enabled = False
    Hint = 'Italic'
    ImageIndex = 29
    ShortCut = 16457
  end
  object RichEditUnderline1: TRichEditUnderline
    Category = 'Format'
    AutoCheck = True
    Caption = '&Underline'
    Enabled = False
    Hint = 'Underline'
    ImageIndex = 28
    ShortCut = 16469
  end
  object RichEditStrikeOut1: TRichEditStrikeOut
    Category = 'Format'
    AutoCheck = True
    Caption = '&Strikeout'
    Enabled = False
    Hint = 'Strikeout'
  end
  object RichEditBullets1: TRichEditBullets
    Category = 'Format'
    AutoCheck = True
    Caption = '&Bullets'
    Enabled = False
    Hint = 'Bullets|Inserts a bullet on the current line'
    ImageIndex = 38
  end
  object RichEditAlignLeft1: TRichEditAlignLeft
    Category = 'Format'
    AutoCheck = True
    Caption = 'Align &Left'
    Enabled = False
    Hint = 'Align Left|Aligns text at the left indent'
    ImageIndex = 35
  end
```

```
    Caption = 'Standard Style'
    GroupIndex = 1
    OnExecute =
StdStyleActnExecute
  end
  object XPStyleActn: TAction
    Category = 'Style'
    AutoCheck = True
    Caption = 'XP Style'
    GroupIndex = 1
    OnExecute =
XPStyleActnExecute
  end
  object ShadowActn: TAction
    Category = 'Style'
    AutoCheck = True
    Caption = 'Menu Shadows'
    OnExecute =
ShadowActnExecute
  end
  object FilePageSetup1:
TFilePageSetup
    Category = 'File'
    Caption = 'FilePageSetup1'
    Dialog.MinMarginLeft = 0
    Dialog.MinMarginTop = 0
    Dialog.MinMarginRight = 0
    Dialog.MinMarginBottom = 0
    Dialog.MarginLeft = 1000
    Dialog.MarginTop = 1000
    Dialog.MarginRight = 1000
    Dialog.MarginBottom = 1000
    Dialog.PageWidth = 8500
    Dialog.PageHeight = 11000
  end
 end
 object ImageList1: TImageList
  Left = 50
  Top = 137
  Bitmap = {
    000000000000}
 end
End

\\\\\\\\\\\\\\\\
```

ObjectS

```
    Caption = 'E&xit'
    Hint = 'Exit|Quits the
application'
    ImageIndex = 43
  end
  object SearchFind1: TSearchFind
    Category = 'Search'
    Caption = '&Find...'
    Hint = 'Find|Finds the specified
text'
    ImageIndex = 34
    ShortCut = 16454
  end
  object SearchFindNext1:
TSearchFindNext
    Category = 'Search'
    Caption = 'Find &Next'
    Enabled = False
    Hint = 'Find Next|Repeats the
last find'
    ImageIndex = 33
    ShortCut = 114
  end
  object SearchReplace1:
TSearchReplace
    Category = 'Search'
    Caption = '&Replace'
    Hint = 'Replace|Replaces specific
text with different text'
    ImageIndex = 32
  end
  object SearchFindFirst1:
TSearchFindFirst
    Category = 'Search'
    Caption = 'F&ind First'
    Hint = 'Find First|Finds the first
occurance of specified text'
  end
  object CustomizeActionBars1:
TCustomizeActionBars
    Category = 'Tools'
    Caption = '&Customize'
    CustomizeDlg.StayOnTop =
False
  end
  object StdStyleActn: TAction
    Category = 'Style'
    AutoCheck = True
```

الملف التنفيذي لبرنامج

(ActionBands)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
ActionBands
object Form1: TForm1
  Left = 202
  Top = 114
  Width = 565
  Height = 368
  Caption = 'ActionBars Demo'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = False
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  object ActionMainMenuBar1: TActionMainMenuBar
    Left = 0
    Top = 0
    Width = 557
    Height = 24
    UseSystemFont = False
    ActionManager = ActionManager1
    Caption = 'ActionMainMenuBar1'
    ColorMap.HighlightColor = clBtnHighlight
    ColorMap.UnusedColor = 15988985
    ColorMap.SelectedColor = clHighlight
    EdgeBorders = [ebBottom]
    EdgeOuter = esNone
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'Tahoma'
    Font.Style = []
    Spacing = 4
  end
  object ToolActionBar1: TActionToolBar
    Left = 0
    Top = 24
    Width = 557
    Height = 28
    ActionManager = ActionManager1
    Caption = 'ToolActionBar1'
    ColorMap.HighlightColor = clBtnHighlight
    ColorMap.UnusedColor = 15988985
    ColorMap.SelectedColor = clHighlight
    EdgeBorders = [ebLeft, ebTop, ebRight, ebBottom]
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    Spacing = 0
  end
  object ToolActionBar2: TActionToolBar
    Left = 0
    Top = 52
    Width = 557
    Height = 28
    ActionManager = ActionManager1
    Caption = 'ToolActionBar2'
    ColorMap.HighlightColor = clBtnHighlight
    ColorMap.UnusedColor = 15988985
    ColorMap.SelectedColor = clHighlight
    EdgeBorders = [ebLeft, ebTop, ebRight, ebBottom]
```

```
    Width = 557
    Height = 19
    AutoHint = True
    Panels = <>
  end
  object ActionManager1: TActionManager
    ActionBars.SessionCount = 6
    ActionBars = <
      item
        Items = <
          item
            Items = <
              item
                Action = FileOpen1
                ImageIndex = 30
                ShortCut = 16463
              end
              item
                Action = FileSaveAs1
                ImageIndex = 8
              end
              item
                Action = FileRun1
                UsageCount = -1
              end
              item
                Caption = '-'
              end
              item
                Action = FileExit1
                ImageIndex = 43
                UsageCount = -1
              end>
            Caption = '&File'
          end
          item
            Items = <
              item
                Action = EditCut1
                ImageIndex = 0
                ShortCut = 16472
              end
              item
                Action = EditCopy1
                ImageIndex = 1
                ShortCut = 16451
              end
```

```
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    Spacing = 0
  end
  object ToolActionBar3: TActionToolBar
    Left = 0
    Top = 80
    Width = 557
    Height = 28
    ActionManager = ActionManager1
    Caption = 'ToolActionBar3'
    ColorMap.HighlightColor = clBtnHighlight
    ColorMap.UnusedColor = 15988985
    ColorMap.SelectedColor = clHighlight
    EdgeBorders = [ebLeft, ebTop, ebRight, ebBottom]
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clWindowText
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    Spacing = 0
  end
  object RichEdit1: TRichEdit
    Left = 0
    Top = 108
    Width = 557
    Height = 207
    Align = alClient
    BevelKind = bkTile
    BorderStyle = bsNone
    TabOrder = 4
  end
  object StatusBar1: TStatusBar
    Left = 0
    Top = 315
```

```
      end
      item
        Action = RichEditItalic1
        ImageIndex = 29
        ShortCut = 16457
      end
      item
        Action = RichEditUnderline1
        ImageIndex = 28
        ShortCut = 16469
      end
      item
        Action = RichEditStrikeOut1
      end
      item
        Action = RichEditBullets1
        Caption = 'Bull&ets'
        ImageIndex = 38
      end
      item
        Action = RichEditAlignLeft1
        ImageIndex = 35
      end
      item
        Action = RichEditAlignRight1
        ImageIndex = 36
      end
      item
        Action = RichEditAlignCenter1
        ImageIndex = 37
      end>
    Caption = 'F&ormat'
  end
  item
    Items = <
      item
        Action = CustomizeActionBars1
        UsageCount = -1
      end>
    Caption = '&Tools'
  end
  item
```

```
      item
        Action = EditPaste1
        ImageIndex = 2
        ShortCut = 16470
      end
      item
        Action = EditSelectAll1
        ShortCut = 16449
      end
      item
        Action = EditUndo1
        ImageIndex = 3
        ShortCut = 16474
      end
      item
        Action = EditDelete1
        ImageIndex = 5
        ShortCut = 46
      end>
    Caption = '&Edit'
  end
  item
    Items = <
      item
        Action = SearchFind1
        ImageIndex = 34
        ShortCut = 16454
      end
      item
        Action = SearchFindNext1
        ImageIndex = 33
        ShortCut = 114
      end
      item
        Action = SearchReplace1
        ImageIndex = 32
      end
      item
        Action = SearchFindFirst1
      end>
    Caption = '&Search'
  end
  item
    Items = <
      item
        Action = RichEditBold1
        ImageIndex = 31
        ShortCut = 16450
```

```
        Caption = 'Bull&ets'
        ImageIndex = 38
      end
      item
        Action = RichEditAlignLeft1
        ImageIndex = 35
      end
      item
        Action = RichEditAlignRight1
        ImageIndex = 36
      end
      item
        Action = RichEditAlignCenter1
        ImageIndex = 37
      end>
    ActionBar = ToolActionBar1
  end
  item
    Items = <
      item
        Action = SearchFind1
        ImageIndex = 34
        ShortCut = 16454
      end
      item
        Action = SearchFindNext1
        ImageIndex = 33
        ShortCut = 114
      end
      item
        Action = SearchReplace1
        ImageIndex = 32
      end
      item
        Caption = '-'
        LastSession = 6
      end
      item
        Action = SearchFindFirst1
      end>
    ActionBar = ToolActionBar2
    AutoSize = False
  end
  item
    Items = <
      item
```

```
        Items = <
          item
            Action = StdStyleActn
            Caption = '&Standard Style'
            LastSession = 6
          end
          item
            Action = XPStyleActn
            Caption = '&XP Style'
            LastSession = 6
          end
          item
            Caption = '-'
          end
          item
            Action = ShadowActn
            Caption = '&Menu Shadows'
            LastSession = 6
          end>
        Caption = 'Sty&le'
      end>
    ActionBar = ActionMainMenuBar1
  end
  item
    Items = <
      item
        Action = RichEditBold1
        ImageIndex = 31
        ShortCut = 16450
      end
      item
        Action = RichEditItalic1
        ImageIndex = 29
        ShortCut = 16457
      end
      item
        Action = RichEditUnderline1
        ImageIndex = 28
        ShortCut = 16469
      end
      item
        Action = RichEditStrikeOut1
      end
      item
        Action = RichEditBullets1
```

```
    Caption = '&Copy'
    Enabled = False
    Hint = 'Copy|Copies the selection and puts it on the Clipboard'
    ImageIndex = 1
    ShortCut = 16451
  end
  object EditPaste1: TEditPaste
    Category = 'Edit'
    Caption = '&Paste'
    Enabled = False
    Hint = 'Paste|Inserts Clipboard contents'
    ImageIndex = 2
    ShortCut = 16470
  end
  object EditSelectAll1: TEditSelectAll
    Category = 'Edit'
    Caption = 'Select &All'
    Enabled = False
    Hint = 'Select All|Selects the entire document'
    ShortCut = 16449
  end
  object EditUndo1: TEditUndo
    Category = 'Edit'
    Caption = '&Undo'
    Enabled = False
    Hint = 'Undo|Reverts the last action'
    ImageIndex = 3
    ShortCut = 16474
  end
  object EditDelete1: TEditDelete
    Category = 'Edit'
    Caption = '&Delete'
    Enabled = False
    Hint = 'Delete|Erases the selection'
    ImageIndex = 5
    ShortCut = 46
  end
  object RichEditBold1: TRichEditBold
    Category = 'Format'
    AutoCheck = True
    Caption = '&Bold'
```

```
          Action = EditCut1
          ImageIndex = 0
          ShortCut = 16472
        end
        item
          Action = EditCopy1
          ImageIndex = 1
          ShortCut = 16451
        end
        item
          Action = EditPaste1
          ImageIndex = 2
          ShortCut = 16470
        end
        item
          Action = EditSelectAll1
          ShortCut = 16449
        end
        item
          Action = EditUndo1
          ImageIndex = 3
          ShortCut = 16474
        end
        item
          Action = EditDelete1
          ImageIndex = 5
          ShortCut = 46
        end>
      ActionBar = ToolActionBar3
      AutoSize = False
    end>
  Images = ImageList1
  OnStateChange = ActionManager1StateChange
  Left = 125
  Top = 138
  StyleName = 'Standard'
  object EditCut1: TEditCut
    Category = 'Edit'
    Caption = 'Cu&t'
    Enabled = False
    Hint = 'Cut|Cuts the selection and puts it on the Clipboard'
    ImageIndex = 0
    ShortCut = 16472
  end
  object EditCopy1: TEditCopy
    Category = 'Edit'
```

```
    Enabled = False
    Hint = 'Align Left|Aligns text at the left indent'
    ImageIndex = 35
  end
  object RichEditAlignRight1: TRichEditAlignRight
    Category = 'Format'
    AutoCheck = True
    Caption = 'Align &Right'
    Enabled = False
    Hint = 'Align Right|Aligns text at the right indent'
    ImageIndex = 36
  end
  object RichEditAlignCenter1: TRichEditAlignCenter
    Category = 'Format'
    AutoCheck = True
    Caption = '&Center'
    Enabled = False
    Hint = 'Center|Centers text between margins'
    ImageIndex = 37
  end
  object FileOpen1: TFileOpen
    Category = 'File'
    Caption = '&Open...'
    Hint = 'Open|Opens an existing file'
    ImageIndex = 30
    ShortCut = 16463
    OnAccept = FileOpen1Accept
  end
  object FileSaveAs1: TFileSaveAs
    Category = 'File'
    Caption = 'Save &As...'
    Hint = 'Save As|Saves the active file with a new name'
    ImageIndex = 8
    OnAccept = FileSaveAs1Accept
  end
  object FileRun1: TFileRun
    Category = 'File'
    Browse = False
    BrowseDlg.Title = 'Run'
    Caption = '&Run...'
    Hint = 'Run|Runs an applicatoin'
```

```
    Enabled = False
    Hint = 'Bold'
    ImageIndex = 31
    ShortCut = 16450
  end
  object RichEditItalic1: TRichEditItalic
    Category = 'Format'
    AutoCheck = True
    Caption = '&Italic'
    Enabled = False
    Hint = 'Italic'
    ImageIndex = 29
    ShortCut = 16457
  end
  object RichEditUnderline1: TRichEditUnderline
    Category = 'Format'
    AutoCheck = True
    Caption = '&Underline'
    Enabled = False
    Hint = 'Underline'
    ImageIndex = 28
    ShortCut = 16469
  end
  object RichEditStrikeOut1: TRichEditStrikeOut
    Category = 'Format'
    AutoCheck = True
    Caption = '&Strikeout'
    Enabled = False
    Hint = 'Strikeout'
  end
  object RichEditBullets1: TRichEditBullets
    Category = 'Format'
    AutoCheck = True
    Caption = '&Bullets'
    Enabled = False
    Hint = 'Bullets|Inserts a bullet on the current line'
    ImageIndex = 38
  end
  object RichEditAlignLeft1: TRichEditAlignLeft
    Category = 'Format'
    AutoCheck = True
    Caption = 'Align &Left'
```

```
False
   end
   object StdStyleActn: TAction
     Category = 'Style'
     AutoCheck = True
     Caption = 'Standard Style'
     GroupIndex = 1
     OnExecute =
StdStyleActnExecute
   end
   object XPStyleActn: TAction
     Category = 'Style'
     AutoCheck = True
     Caption = 'XP Style'
     GroupIndex = 1
     OnExecute =
XPStyleActnExecute
   end
   object ShadowActn: TAction
     Category = 'Style'
     AutoCheck = True
     Caption = 'Menu Shadows'
     OnExecute =
ShadowActnExecute
   end
   object FilePageSetup1:
TFilePageSetup
     Category = 'File'
     Caption = 'FilePageSetup1'
     Dialog.MinMarginLeft = 0
     Dialog.MinMarginTop = 0
     Dialog.MinMarginRight = 0
     Dialog.MinMarginBottom = 0
     Dialog.MarginLeft = 1000
     Dialog.MarginTop = 1000
     Dialog.MarginRight = 1000
     Dialog.MarginBottom = 1000
     Dialog.PageWidth = 8500
     Dialog.PageHeight = 11000
   end
  end
  object ImageList1: TImageList
   Left = 50
   Top = 137
   Bitmap = {
     000000000000}
  end
End
```

```
     Operation = 'open'
     ShowCmd = scShowNormal
   end
   object FileExit1: TFileExit
     Category = 'File'
     Caption = 'E&xit'
     Hint = 'Exit|Quits the
application'
     ImageIndex = 43
   end
   object SearchFind1: TSearchFind
     Category = 'Search'
     Caption = '&Find...'
     Hint = 'Find|Finds the specified
text'
     ImageIndex = 34
     ShortCut = 16454
   end
   object SearchFindNext1:
TSearchFindNext
     Category = 'Search'
     Caption = 'Find &Next'
     Enabled = False
     Hint = 'Find Next|Repeats the
last find'
     ImageIndex = 33
     ShortCut = 114
   end
   object SearchReplace1:
TSearchReplace
     Category = 'Search'
     Caption = '&Replace'
     Hint = 'Replace|Replaces specific
text with different text'
     ImageIndex = 32
   end
   object SearchFindFirst1:
TSearchFindFirst
     Category = 'Search'
     Caption = 'F&ind First'
     Hint = 'Find First|Finds the first
occurance of specified text'
   end
   object CustomizeActionBars1:
TCustomizeActionBars
     Category = 'Tools'
     Caption = '&Customize'
     CustomizeDlg.StayOnTop =
```

Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, ExtCtrls, FrmData, FrmMD;
type
TForm1 = class(TForm)
MDFrame: TMasterDetailFrame;
SimpleFrame: TDataFrame;
Splitter1: TSplitter;
<------>procedure FormCreate(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;
var
Form1: TForm1;
implementation
{$R *.dfm}
<------>procedure TForm1.FormCreate(Sender: TObject);
begin
with SimpleFrame, Table1 do begin
TableName := 'Biolife';
with FancyFrame1 do begin
DBMemo1.DataSource := DataSource1;
DBMemo1.DataField := 'Notes';
DBImage1.DataSource := DataSource1;
DBImage1.DataField := 'Graphic';
end;
Open;
end;
with MDFrame do begin
with MasterFrame, Table1 do begin
FancyFrame1.Free;
TableName := 'Customer';

\\\\\\\\\\\\\\\\\

Procedure2
الملف التنفيذي لبرنامج
(Db)
إعداد **علاء الدين اللباد**
هاتف ٠٩٤٤٥٧٥٣٧١
علاء الدين اللباد ٠٩٤٤٥٧٥٣٧١
الملف التنفيذي للبرنامج DB
يوجد فيه اجراءان فقط
Procedures 2
program FramesDemo;
uses
Forms,
FrmData in 'FrmData.pas' {DataFrame: TFrame},
FrmFancy in 'FrmFancy.pas' {FancyFrame: TFrame},
FrmMD in 'FrmMD.pas' {MasterDetailFrame: TFrame},
FrmMain in 'FrmMain.pas' {Form1};
{$R *.RES}
begin
Application.Initialize;
Application.CreateForm(TForm1, Form1);
Application.Run;
end.
unit FrmMain;
interface
uses

```
unit FrmData;

interface

uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, DBCtrls, Db, DBTables, Grids, DBGrids, FrmFancy;

type
  TDataFrame = class(TFrame)
    DBGrid1: TDBGrid;
    DataSource1: TDataSource;
    Table1: TTable;
    DBNavigator1: TDBNavigator;
    FancyFrame1: TFancyFrame;
    Splitter1: TSplitter;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}

end.


\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\\\\\\\\\\\\\\\\\\\\\\

unit FrmFancy;

interface

uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, DBCtrls, StdCtrls;

type
  TFancyFrame = class(TFrame)
    DBMemo1: TDBMemo;
```

```
    Open;
   end;
   with DetailFrame, Table1 do begin
    FancyFrame1.Free;
    MasterSource := MasterFrame.DataSource1;
    MasterFields := 'CustNo';
    IndexName := 'CustNo';
    TableName := 'Orders';
    Open;
   end;
  end;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

unit FrmMD;

interface

uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, FrmData;

type
  TMasterDetailFrame = class(TFrame)
    MasterFrame: TDataFrame;
    DetailFrame: TDataFrame;
    Splitter2: TSplitter;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}

end.


\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

```
 end
  inline MDFrame: TMasterDetailFrame
    Left = 0
    Top = 297
    Width = 775
    Height = 263
    Align = alBottom
    TabOrder = 0
    inherited Splitter2: TSplitter
      Height = 263
    end
    inherited MasterFrame: TDataFrame
      Height = 263
      inherited Splitter1: TSplitter
        Top = 159
      end
      inherited DBGrid1: TDBGrid
        Height = 134
      end
      inherited DBNavigator1: TDBNavigator
        Hints.Strings = ()
      end
      inherited FancyFrame1: TFancyFrame
        Top = 162
      end
    end
    inherited DetailFrame: TDataFrame
      Width = 434
      Height = 263
      inherited Splitter1: TSplitter
        Top = 159
        Width = 434
      end
      inherited DBGrid1: TDBGrid
        Width = 434
        Height = 134
      end
      inherited DBNavigator1: TDBNavigator
        Width = 434
        Hints.Strings = ()
      end
      inherited FancyFrame1:
```

```
    DBImage1: TDBImage;
    Splitter1: TSplitter;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

Objects 15

الملف النصي لبرنامج

(DB)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
object Form1: TForm1
  Left = 209
  Top = 158
  Width = 783
  Height = 594
  Caption = 'Form1'
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = False
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
<-------->  object Splitter1: TSplitter
    Left = 0
    Top = 294
    Width = 775
    Height = 3
    Cursor = crVSplit
    Align = alBottom
```

object MasterDetailFrame: TMasterDetailFrame
Left = 0
Top = 0
Width = 553
Height = 367
TabOrder = 0
<-------> object Splitter2: TSplitter
Left = 338
Top = 0
Height = 367
end
inline MasterFrame: TDataFrame
Left = 0
Top = 0
Width = 338
Height = 367
Align = alLeft
TabOrder = 0
inherited Splitter1: TSplitter
Top = 263
end
inherited DBGrid1: TDBGrid
Height = 238
end
inherited DBNavigator1: TDBNavigator
Hints.Strings = ()
end
inherited FancyFrame1: TFancyFrame
Top = 266
end
end
inline DetailFrame: TDataFrame
Left = 341
Top = 0
Width = 212
Height = 367
Align = alClient
TabOrder = 1
inherited Splitter1: TSplitter
Top = 263
Width = 212
end
inherited DBGrid1: TDBGrid
Width = 212
Height = 238

TFancyFrame
Top = 162
Width = 434
inherited DBImage1: TDBImage
Width = 270
end
end
end
end
inline SimpleFrame: TDataFrame
Left = 0
Top = 0
Width = 775
Height = 294
Align = alClient
TabOrder = 1
inherited Splitter1: TSplitter
Top = 190
Width = 775
end
inherited DBGrid1: TDBGrid
Width = 775
Height = 165
end
inherited DBNavigator1: TDBNavigator
Width = 775
Hints.Strings = ()
end
inherited FancyFrame1: TFancyFrame
Top = 193
Width = 775
inherited Splitter1: TSplitter
Left = 489
end
inherited DBMemo1: TDBMemo
Width = 489
end
inherited DBImage1: TDBImage
Left = 492
Width = 283
end
end
end
End

```
   Left = 0
   Top = 0
   Width = 338
   Height = 25
   DataSource = DataSource1
   Align = alTop
   TabOrder = 1
  end
  inline FancyFrame1: TFancyFrame
   Left = 0
   Top = 260
   Width = 338
   Height = 101
   Align = alBottom
   TabOrder = 2
   inherited DBImage1: TDBImage
    Width = 174
   end
  end
 <-------> object DataSource1:
TDataSource
   DataSet = Table1
   Left = 40
   Top = 200
  end
 <-------> object Table1: TTable
   DatabaseName = 'DBDEMOS'
   Left = 8
   Top = 200
  end
End
object FancyFrame: TFancyFrame
  Left = 0
  Top = 0
  Width = 320
  Height = 101
  TabOrder = 0
 <-------> object Splitter1: TSplitter
   Left = 161
   Top = 0
   Height = 101
  end
 <-------> object DBMemo1:
TDBMemo
   Left = 0
   Top = 0
   Width = 161
   Height = 101
```

```
   end
   inherited DBNavigator1:
TDBNavigator
     Width = 212
     Hints.Strings = ()
   end
   inherited FancyFrame1:
TFancyFrame
     Top = 266
     Width = 212
     inherited DBImage1: TDBImage
       Width = 48
     end
   end
  end
End

object DataFrame: TDataFrame
  Left = 0
  Top = 0
  Width = 338
  Height = 361
  TabOrder = 0
 <-------> object Splitter1: TSplitter
   Left = 0
   Top = 257
   Width = 338
   Height = 3
   Cursor = crVSplit
   Align = alBottom
  end
 <-------> object DBGrid1: TDBGrid
   Left = 0
   Top = 25
   Width = 338
   Height = 232
   Align = alClient
   DataSource = DataSource1
   TabOrder = 0
   TitleFont.Charset =
DEFAULT_CHARSET
   TitleFont.Color = clWindowText
   TitleFont.Height = -11
   TitleFont.Name = 'MS Sans Serif'
   TitleFont.Style = []
  end
 <-------> object DBNavigator1:
TDBNavigator
```

```
  BtnTrans: TButton;
<--------> procedure BtnCloseClick(Sender: TObject);
<--------> procedure BtnViewsClick(Sender: TObject);
<--------> procedure BtnTriggClick(Sender: TObject);
<--------> procedure BtnQrySPClick(Sender: TObject);
<--------> procedure BtnExecSPClick(Sender: TObject);
<--------> procedure BtnTransClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FrmLauncher: TFrmLauncher;

implementation

{$R *.dfm}

uses
  FrmExeSP,   { The Executable Stored Procedure Demo }
  FrmQrySP,   { The Query Stored Procedure demo }
  FrmTrans,   { The Transaction Editing demo }
  FrmTrigg,   { The Trigger Demo }
  FrmViews,   { The View Demo }
  Variants;

procedure TFrmLauncher.BtnCloseClick(Sender: TObject);
begin
  Close;
end;

procedure TFrmLauncher.BtnViewsClick(Sender: TObject);
begin
```

```
    Align = alLeft
    TabOrder = 0
  end
<-------> object DBImage1: TDBImage
    Left = 164
    Top = 0
    Width = 156
    Height = 101
    Align = alClient
    TabOrder = 1
  end
End
```

الملف التنفيذي للبرنامج ويحتوي على ٣٢ إجراءا (Procedure32) ويعتبر من البرامج الشاملة المرتبطة بقواعد البيانات وهو برنامج مفتوح الشيفرة وموجود ضمن دلفي ٧ باسم CsDemos

Procedure32
الملف التنفيذي لبرنامج
(CsDemos)
إعداد **علاء الدين اللباد**
هاتف ٠٩٤٤٥٧٥٣٧١

```
unit Frmmain;

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, DB, StdCtrls;

type
  TFrmLauncher = class(TForm)
    BtnTrigg: TButton;
    BtnViews: TButton;
    BtnQrySP: TButton;
    BtnExecSP: TButton;
    BtnClose: TButton;
```

class(TDataModule)
SalesTable: TTable;
SalesSource: TDataSource;
CustomerTable: TTable;
CustomerSource: TDataSource;
ShipOrderProc: TStoredProc;
EmployeeDatabase: TDatabase;
EmployeeSource: TDataSource;
EmployeeTable: TTable;
EmployeeTableEMP_NO: TSmallintField;
EmployeeTableFIRST_NAME: TStringField;
EmployeeTableLAST_NAME: TStringField;
EmployeeTablePHONE_EXT: TStringField;
EmployeeTableHIRE_DATE: TDateTimeField;
EmployeeTableDEPT_NO: TStringField;
EmployeeTableJOB_CODE: TStringField;
EmployeeTableJOB_GRADE: TSmallintField;
EmployeeTableJOB_COUNTRY: TStringField;
EmployeeTableSALARY: TFloatField;
EmployeeTableFULL_NAME: TStringField;
DeleteEmployeeProc: TStoredProc;
SalaryHistoryTable: TTable;
SalaryHistorySource: TDataSource;
SalaryHistoryTableCHANGE_DATE: TDateTimeField;
SalaryHistoryTableUPDATER_ID: TStringField;
SalaryHistoryTableOLD_SALARY: TFloatField;
SalaryHistoryTablePERCENT_CHANGE: TFloatField;

FrmViewDemo.ShowModal;
end;
procedure TFrmLauncher.BtnTriggClick(Sender: TObject);
begin
FrmTriggerDemo.ShowModal;
end;
procedure TFrmLauncher.BtnQrySPClick(Sender: TObject);
begin
FrmQueryProc.ShowModal;
end;
procedure TFrmLauncher.BtnExecSPClick(Sender: TObject);
begin
FrmExecProc.ShowModal;
end;
procedure TFrmLauncher.BtnTransClick(Sender: TObject);
begin
FrmTransDemo.ShowModal;
end;
end.
\\\\\\\\\\\\\\\\\\\\\\\\
unit DmCSDemo;
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
DBTables, DB;
type
TDmEmployee =

```
TIntegerField;
    SalesTableDISCOUNT: TFloatField;
    SalesTableITEM_TYPE: TStringField;
    SalesTableAGED: TFloatField;
    EmployeeLookup: TTable;
    SmallintField1: TSmallintField;
    StringField1: TStringField;
    StringField2: TStringField;
    StringField3: TStringField;
    DateTimeField1: TDateTimeField;
    StringField4: TStringField;
    StringField5: TStringField;
    SmallintField2: TSmallintField;
    StringField6: TStringField;
    FloatField1: TFloatField;
    StringField7: TStringField;
    SalaryHistoryTableEMP_NO: TSmallintField;
    SalaryHistoryTableEMPLOYEE: TStringField;
<--------> procedure EmployeeTableBeforeDelete(DataSet: TDataSet);
<--------> procedure EmployeeTableAfterPost(DataSet: TDataSet);
<--------> procedure DmEmployeeCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  DmEmployee: TDmEmployee;

implementation

{$R *.dfm}

{ Note: Business rules go in the data model.  Here is an example, used by
  the transaction editing demo.
```

```
SalaryHistoryTableNEW_SALARY: TFloatField;
    CustomerTableCUST_NO: TIntegerField;
    CustomerTableCUSTOMER: TStringField;

CustomerTableCONTACT_FIRST: TStringField;

CustomerTableCONTACT_LAST: TStringField;
    CustomerTablePHONE_NO: TStringField;

CustomerTableADDRESS_LINE1: TStringField;

CustomerTableADDRESS_LINE2: TStringField;
    CustomerTableCITY: TStringField;

CustomerTableSTATE_PROVINCE: TStringField;
    CustomerTableCOUNTRY: TStringField;
    CustomerTablePOSTAL_CODE: TStringField;
    CustomerTableON_HOLD: TStringField;
    SalesTablePO_NUMBER: TStringField;
    SalesTableCUST_NO: TIntegerField;
    SalesTableSALES_REP: TSmallintField;
    SalesTableORDER_STATUS: TStringField;
    SalesTableORDER_DATE: TDateTimeField;
    SalesTableSHIP_DATE: TDateTimeField;
    SalesTableDATE_NEEDED: TDateTimeField;
    SalesTablePAID: TStringField;
    SalesTableQTY_ORDERED: TIntegerField;
    SalesTableTOTAL_VALUE:
```

```
  with SalaryHistoryTable do if Active then Refresh;
end;

procedure TDmEmployee.DmEmployeeCreate(Sender: TObject);
begin
  EmployeeDatabase.Open;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

unit Frmexesp;

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls,
  StdCtrls, Forms, DBCtrls, DB, DBGrids, Buttons, DBTables, Mask, Grids,
  ExtCtrls;

type
  TFrmExecProc = class(TForm)
    DBGrid1: TDBGrid;
    ScrollBox: TScrollBox;
    Label1: TLabel;
    EditCUST_NO: TDBEdit;
    Label2: TLabel;
    EditCUSTOMER: TDBEdit;
    Label3: TLabel;
    EditCONTACT_FIRST: TDBEdit;
    EditCONTACT_LAST: TDBEdit;
    EditPHONE_NO: TDBEdit;
    Label6: TLabel;
    EditADDRESS_LINE: TDBEdit;
    EditADDRESS_LINE2: TDBEdit;
    EditCITY: TDBEdit;
    EditSTATE_PROVINCE: TDBEdit;
    EditCOUNTRY: TDBEdit;
    EditPOSTAL_CODE: TDBEdit;
```

```
Deletes for the employee table are done
  with a stored procedure rather than the normal BDE record delete
  mechanism, so an audit trail could be provided, etc... }

{ The database, EmployeeDatabase, is the InterBase example EMPLOYEE.GDB database
  accessed thru the BDE alias IBLOCAL.  This database contains examples
  of stored procedures, triggers, check constraints, views, etc., many of
  which are used within this demo project. }

procedure TDmEmployee.EmployeeTableBeforeDelete(DataSet: TDataSet);
begin
  { Assign the current employee's id to the stored procedure's parameter }
  DeleteEmployeeProc.Params.ParamValues['EMP_NUM'] := EmployeeTable['EMP_NO'];
  DeleteEmployeeProc.ExecProc;        { Trigger the stored proc }
  EmployeeTable.Refresh;              { Refresh the data }
  { Block the EmployeeTable delete since the stored procedure did the work }
  Abort;
end;

procedure TDmEmployee.EmployeeTableAfterPost(DataSet: TDataSet);
begin
  { A change in an employee salary triggers a change in the salary history,
    so if that table is open, it needs to be refreshed now }
```

```
TObject);
begin
  { Disable DataEvents from the
SalesTable for this form now }
  SalesSource.Enabled := False;
end;

procedure
TFrmExecProc.SalesSourceDataCha
nge(Sender: TObject; Field: TField);
begin
  if
DmEmployee.SalesTable['ORDER_
STATUS'] <> NULL then
  BtnShipOrder.Enabled :=

AnsiCompareText(DmEmployee.Sal
esTable['ORDER_STATUS'],
'SHIPPED') <> 0;
end;

procedure
TFrmExecProc.BtnShipOrderClick(
Sender: TObject);
begin
  with DmEmployee do
  begin

ShipOrderProc.Params[0].AsString
:= SalesTable['PO_NUMBER'];
    ShipOrderProc.ExecProc;
    SalesTable.Refresh;
  end;
end;

end.



\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

unit Frmmain;

interface

uses
  SysUtils, Windows, Messages,
Classes, Graphics, Controls,
```

```
    DBNavigator: TDBNavigator;
    Panel1: TPanel;
    Panel3: TPanel;
    Panel2: TPanel;
    DBCheckBox1: TDBCheckBox;
    Label4: TLabel;
    BtnShipOrder: TSpeedButton;
    BitBtn1: TBitBtn;
    SalesSource: TDataSource;
<--------> procedure
SalesSourceDataChange(Sender:
TObject; Field: TField);
<--------> procedure
BtnShipOrderClick(Sender:
TObject);
<--------> procedure
FormShow(Sender: TObject);
<--------> procedure
FormHide(Sender: TObject);
  private
    { private declarations }
  public
    { public declarations }
  end;

var
  FrmExecProc: TFrmExecProc;

implementation

uses DmCSDemo, Variants;

{$R *.dfm}

procedure
TFrmExecProc.FormShow(Sender:
TObject);
begin
  DmEmployee.SalesTable.Open;

DmEmployee.CustomerTable.Open;
  { Enable DataEvents from the
SalesTable for this form now }
  SalesSource.Enabled := True;
end;

procedure
TFrmExecProc.FormHide(Sender:
```

```
 Forms, Dialogs, DB, StdCtrls;

type
  TFrmLauncher = class(TForm)
    BtnTrigg: TButton;
    BtnViews: TButton;
    BtnQrySP: TButton;
    BtnExecSP: TButton;
    BtnClose: TButton;
    BtnTrans: TButton;
<--------> procedure BtnCloseClick(Sender: TObject);
<--------> procedure BtnViewsClick(Sender: TObject);
<--------> procedure BtnTriggClick(Sender: TObject);
<--------> procedure BtnQrySPClick(Sender: TObject);
<--------> procedure BtnExecSPClick(Sender: TObject);
<--------> procedure BtnTransClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FrmLauncher: TFrmLauncher;

implementation

{$R *.dfm}

uses
   FrmExeSP,  { The Executable Stored Procedure Demo }
   FrmQrySP,  { The Query Stored Procedure demo }
   FrmTrans,  { The Transaction Editing demo }
   FrmTrigg,  { The Trigger Demo }
   FrmViews,  { The View Demo }
   Variants;

procedure TFrmLauncher.BtnCloseClick(Sender: TObject);
begin
  Close;
end;

procedure TFrmLauncher.BtnViewsClick(Sender: TObject);
begin
  FrmViewDemo.ShowModal;
end;

procedure TFrmLauncher.BtnTriggClick(Sender: TObject);
begin
  FrmTriggerDemo.ShowModal;
end;

procedure TFrmLauncher.BtnQrySPClick(Sender: TObject);
begin
  FrmQueryProc.ShowModal;
end;

procedure TFrmLauncher.BtnExecSPClick(Sender: TObject);
begin
  FrmExecProc.ShowModal;
end;

procedure TFrmLauncher.BtnTransClick(Sender: TObject);
begin
  FrmTransDemo.ShowModal;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\


unit Frmqrysp;

interface
```

```
TFrmQueryProc.WriteMsg(StrWrite: String);
begin
  StatusBar1.SimpleText := StrWrite;
end;

procedure TFrmQueryProc.FormShow(Sender : TObject);
begin
  DmEmployee.EmployeeTable.Open;
  { Allow data flow from the EmployeeTable to the local EmployeeSource.  This
    will allow DataChange events to execute the query procedure }
  EmployeeSource.Enabled := True;
  { Explicit query preparation is not required, but gives the best possible
    performance }
  with EmployeeProjectsQuery do if not Active then Prepare;
end;

procedure TFrmQueryProc.EmployeeDataChange(Sender: TObject; Field: TField);
begin
  { Execute the ProjectsQuery, which uses a query procedure }
  EmployeeProjectsQuery.Close;
  EmployeeProjectsQuery.Params[0].AsInteger :=
    DmEmployee.EmployeeTableEmp_No.Value;
  EmployeeProjectsQuery.Open;

  WriteMsg('Employee ' + DmEmployee.EmployeeTableEmp_No.AsString +
          ' is assigned to ' + IntToStr(EmployeeProjectsQuery.RecordCount) +
          ' project(s).');
end;
```

```

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls,
  StdCtrls, Forms, DBCtrls, DB, DBGrids, DBTables, Grids, ExtCtrls,
  Dialogs, Buttons, Mask, ComCtrls;

type
  TFrmQueryProc = class(TForm)
    DBGrid1: TDBGrid;
    DBGrid2: TDBGrid;
    DBNavigator: TDBNavigator;
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    EmployeeProjectsQuery: TQuery;
    EmployeeProjectsSource: TDataSource;
    StatusBar1: TStatusBar;
    BitBtn1: TBitBtn;
    EmployeeSource: TDataSource;
<--------> procedure EmployeeDataChange(Sender: TObject; Field: TField);
<--------> procedure FormShow(Sender: TObject);
<--------> procedure FormHide(Sender: TObject);
  private
    { private declarations }
<--------> procedure writeMsg( strWrite : String );
  public
    { public declarations }
  end;

var
  FrmQueryProc: TFrmQueryProc;

implementation

uses DmCSDemo;

{$R *.dfm}

procedure
```

```
   { private declarations }
  public
   { public declarations }
  end;

var
  FrmTransDemo:
TFrmTransDemo;

implementation

uses DmCSDemo;

{$R *.dfm}

procedure
TFrmTransDemo.FormShow(Sende
r: TObject);
begin

DmEmployee.EmployeeDatabase.St
artTransaction;

DmEmployee.EmployeeTable.Open;
end;

procedure
TFrmTransDemo.FormHide(Sender
: TObject);
begin

DmEmployee.EmployeeDatabase.Co
mmit;
end;

procedure
TFrmTransDemo.BtnCommitEdits
Click(Sender: TObject);
begin
  if
DmEmployee.EmployeeDatabase.In
Transaction and
     (MessageDlg('Are you sure you
want to commit your changes?',
        mtConfirmation, [mbYes,
mbNo], 0) = mrYes) then
  begin

DmEmployee.EmployeeDatabase.Co
```

```

procedure
TFrmQueryProc.FormHide(Sender:
TObject);
begin
  { Turn off the DataChange event
for our form, since
DmEmployee.EmployeeTable
    is used elsewhere }
  EmployeeSource.Enabled := False;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\

unit Frmtrans;

interface

uses
  SysUtils, Windows, Messages,
Classes, Graphics, Controls,
  StdCtrls, Forms, DBCtrls, DB,
DBGrids, Buttons, DBTables, Grids,
ExtCtrls,
  Dialogs, BDE;

type
  TFrmTransDemo = class(TForm)
    DBGrid1: TDBGrid;
    DBNavigator: TDBNavigator;
    Panel1: TPanel;
    Panel2: TPanel;
    BitBtn1: TBitBtn;
    BtnUndoEdits: TSpeedButton;
    BtnCommitEdits: TSpeedButton;
<--------> procedure
BtnCommitEditsClick(Sender:
TObject);
<--------> procedure
BtnUndoEditsClick(Sender:
TObject);
<--------> procedure
FormShow(Sender: TObject);
<--------> procedure
FormHide(Sender: TObject);
  private
```

```
uses
  SysUtils, Windows, Messages,
Classes, Graphics, Controls,
  StdCtrls, Forms, DBCtrls, DB,
DBGrids, Buttons, DBTables, Grids,
ExtCtrls;

type
  TFrmTriggerDemo = class(TForm)
    DBGrid1: TDBGrid;
    DBGrid2: TDBGrid;
    DBNavigator: TDBNavigator;
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    BitBtn1: TBitBtn;
<-------->  procedure
FormShow(Sender: TObject);
  private
    { private declarations }
  public
    { public declarations }
  end;

var
  FrmTriggerDemo:
TFrmTriggerDemo;

implementation

uses DmCSDemo;

{$R *.dfm}

procedure
TFrmTriggerDemo.FormShow(Send
er: TObject);
begin

DmEmployee.EmployeeTable.Open;

DmEmployee.SalaryHistoryTable.O
pen;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

```
mmit;

DmEmployee.EmployeeDatabase.St
artTransaction;

DmEmployee.EmployeeTable.Refres
h;
  end else
    MessageDlg('Can''t Commit
Changes: No Transaction Active',
mtError, [mbOk], 0);
end;

procedure
TFrmTransDemo.BtnUndoEditsClic
k(Sender: TObject);
begin
  if
DmEmployee.EmployeeDatabase.In
Transaction and
    (MessageDlg('Are you sure you
want to undo all changes made
during the ' +
      'current transaction?',
mtConfirmation, [mbYes, mbNo], 0)
= mrYes) then
  begin

DmEmployee.EmployeeDatabase.Ro
llback;

DmEmployee.EmployeeDatabase.St
artTransaction;

DmEmployee.EmployeeTable.Refres
h;
  end else
    MessageDlg('Can''t Undo Edits:
No Transaction Active', mtError,
[mbOk], 0);
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
unit Frmtrigg;

interface
```

```
unit Frmviews;

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls,
  StdCtrls, Forms, DBCtrls, DB, DBGrids, Buttons, DBTables, Grids, ExtCtrls;

type
  TFrmViewDemo = class(TForm)
    DBGrid1: TDBGrid;
    DBNavigator: TDBNavigator;
    Panel1: TPanel;
    VaryingTableSource: TDataSource;
    Panel2: TPanel;
    VaryingTable: TTable;
    BitBtn1: TBitBtn;
    BtnShowEmployee: TSpeedButton;
    BtnShowPhoneList: TSpeedButton;
<--------->  procedure BtnShowEmployeeClick(Sender: TObject);
<--------->  procedure BtnShowPhoneListClick(Sender: TObject);
<--------->  procedure FormShow(Sender: TObject);
  private
    { private declarations }
<--------->  procedure ShowTable(ATable: string);
  public
    { public declarations }
  end;

var
  FrmViewDemo: TFrmViewDemo;

implementation

{$R *.dfm}
```

```
procedure TFrmViewDemo.ShowTable( ATable: string );
begin
  Screen.Cursor := crHourglass;     { show user something's happening }
  VaryingTable.DisableControls;     { hide data changes from user }
  VaryingTable.Active := FALSE;     { close the table }
  VaryingTable.TableName := ATable;  { update the name }
  VaryingTable.Open;                { open the table }
  VaryingTable.EnableControls;      { paint the changes }
  Screen.Cursor := crDefault;       { reset the pointer }
end;

procedure TFrmViewDemo.FormShow(Sender : TObject);
begin
  VaryingTable.Open;
end;

procedure TFrmViewDemo.BtnShowEmployeeClick(Sender: TObject);
begin
  ShowTable('EMPLOYEE');
end;

procedure TFrmViewDemo.BtnShowPhoneListClick(Sender: TObject);
begin
  ShowTable('PHONE_LIST');
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
ShowHint = True
OnShow = FormShow
PixelsPerInch = 96
TextHeight = 13
<------->object Panel1: TPanel
Left = 0
Top = 0
Width = 390
Height = 41
Align = alTop
TabOrder = 0
<------->object BtnShowEmployee: TSpeedButton
Left = 258
Top = 8
Width = 25
Height = 25
Hint = 'Show Employee Table'
GroupIndex = 1
Down = True
Glyph.Data = {
777777}
NumGlyphs = 2
OnClick = BtnShowEmployeeClick
end
<------->object BtnShowPhoneList: TSpeedButton
Left = 282
Top = 8
Width = 25
Height = 25
Hint = 'Show Phone List View'
GroupIndex = 1
Glyph.Data = {
77777707FF0777777}
NumGlyphs = 2
OnClick = BtnShowPhoneListClick

ObjectS
الملف النصي لبرنامج
(CsDemos)
إعداد **علاء الدين اللباد**
هاتف ٠٩٤٤٥٧٥٣٧١
Objects 139
<------->object FrmViewDemo: TFrmViewDemo
Left = 298
Top = 177
Width = 398
Height = 332
ActiveControl = Panel1
Caption = 'Table/View Switcher'
Color = clBtnFace
Font.Charset =

```
    TitleFont.Color = clBlack
    TitleFont.Height = -11
    TitleFont.Name = 'MS Sans
Serif'
    TitleFont.Style = []
  end
 end
 <------->object
VaryingTableSource: TDataSource
  DataSet = VaryingTable
  Left = 52
  Top = 135
 end
 <------->object VaryingTable:
TTable
  DatabaseName =
'EmployeeDemoDB'
  TableName = 'EMPLOYEE'
  Left = 169
  Top = 119
 end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\\\\\\
<------->object FrmTriggerDemo:
TFrmTriggerDemo
 Left = 684
 Top = 520
 Width = 391
 Height = 342
 Hint = 'Explore the EmployeeTable
to see the trigger sources'
 ActiveControl = Panel1
 Caption = 'Salary Change
Auditing'
 Color = clBtnFace
 Font.Charset =
DEFAULT_CHARSET
 Font.Color = clBlack
 Font.Height = -11
 Font.Name = 'MS Sans Serif'
 Font.Style = []
 OldCreateOrder = True
 Position = poScreenCenter
 ShowHint = True
 OnShow = FormShow
 PixelsPerInch = 96
```

```
  end
  <------->object DBNavigator:
TDBNavigator
   Left = 8
   Top = 8
   Width = 240
   Height = 25
   DataSource =
VaryingTableSource
   Ctl3D = False
   ParentCtl3D = False
   TabOrder = 0
  end
  <------->object BitBtn1: TBitBtn
   Left = 320
   Top = 8
   Width = 60
   Height = 25
   Hint = 'Exit and close this form'
   Caption = 'E&xit'
   TabOrder = 1
   Kind = bkClose
   Style = bsNew
  end
 end
 <------->object Panel2: TPanel
  Left = 0
  Top = 41
  Width = 390
  Height = 257
  Align = alClient
  BevelInner = bvLowered
  BorderWidth = 4
  Caption = 'Panel2'
  TabOrder = 1
  <------->object DBGrid1:
TDBGrid
   Left = 6
   Top = 6
   Width = 378
   Height = 245
   Align = alClient
   BorderStyle = bsNone
   DataSource =
VaryingTableSource
   TabOrder = 0
   TitleFont.Charset =
DEFAULT_CHARSET
```

Top = 6
Width = 371
Height = 125
Hint = 'Changing a salary initiates a trigger'
Align = alClient
BorderStyle = bsNone
DataSource = DmEmployee.EmployeeSource
TabOrder = 0
TitleFont.Charset = DEFAULT_CHARSET
TitleFont.Color = clBlack
TitleFont.Height = -11
TitleFont.Name = 'MS Sans Serif'
TitleFont.Style = []
Columns = <
item
Expanded = False
FieldName = 'FULL_NAME'
ReadOnly = True
Width = 124
Visible = True
end
item
Expanded = False
FieldName = 'SALARY'
Visible = True
end
item
Expanded = False
FieldName = 'JOB_GRADE'
ReadOnly = True
Visible = True
end>
end
end
<-------->object Panel3: TPanel
Left = 0
Top = 178
Width = 383
Height = 130
Align = alClient
BevelInner = bvLowered
BorderWidth = 4
Caption = 'Panel3'
TabOrder = 2

TextHeight = 13
<-------->object Panel1: TPanel
Left = 0
Top = 0
Width = 383
Height = 41
Align = alTop
TabOrder = 0
<-------->object DBNavigator: TDBNavigator
Left = 8
Top = 8
Width = 232
Height = 25
DataSource = DmEmployee.EmployeeSource
VisibleButtons = [nbFirst, nbPrior, nbNext, nbLast, nbEdit, nbPost, nbCancel, nbRefresh]
Ctl3D = False
ParentCtl3D = False
TabOrder = 0
end
<-------->object BitBtn1: TBitBtn
Left = 315
Top = 8
Width = 60
Height = 25
Hint = 'Exit and close this form'
Caption = 'E&xit'
TabOrder = 1
Kind = bkClose
Style = bsNew
end
end
<-------->object Panel2: TPanel
Left = 0
Top = 41
Width = 383
Height = 137
Align = alTop
BevelInner = bvLowered
BorderWidth = 4
Caption = 'Panel2'
TabOrder = 1
<-------->object DBGrid1: TDBGrid
Left = 6

```
'PERCENT_CHANGE'
        Visible = True
      end
      item
        Expanded = False
        FieldName = 'UPDATER_ID'
        Visible = True
      end>
    end
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

<------->object FrmTransDemo: TFrmTransDemo
  Left = 244
  Top = 150
  Width = 506
  Height = 342
  Hint = 'All edits in this form are done in a transaction'
  ActiveControl = Panel1
  Caption = 'Employee Browser'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  ShowHint = True
  OnHide = FormHide
  OnShow = FormShow
  PixelsPerInch = 96
  TextHeight = 13
  <------->object Panel1: TPanel
    Left = 0
    Top = 0
    Width = 498
    Height = 41
    Align = alTop
    TabOrder = 0
    <------->object BtnUndoEdits: TSpeedButton
      Left = 345
```

```
    <------->object DBGrid2: TDBGrid
      Left = 6
      Top = 6
      Width = 371
      Height = 118
      Hint = 'Salary history is maintained by a trigger on salary change'
      Align = alClient
      BorderStyle = bsNone
      DataSource = DmEmployee.SalaryHistorySource
      ReadOnly = True
      TabOrder = 0
      TitleFont.Charset = DEFAULT_CHARSET
      TitleFont.Color = clBlack
      TitleFont.Height = -11
      TitleFont.Name = 'MS Sans Serif'
      TitleFont.Style = []
      Columns = <
        item
          Expanded = False
          FieldName = 'EMPLOYEE'
          Width = 123
          Visible = True
        end
        item
          Expanded = False
          FieldName = 'CHANGE_DATE'
          Visible = True
        end
        item
          Expanded = False
          FieldName = 'OLD_SALARY'
          Visible = True
        end
        item
          Expanded = False
          FieldName = 'NEW_SALARY'
          Visible = True
        end
        item
          Expanded = False
          FieldName =
```

```
  Align = alClient
  BevelInner = bvLowered
  BorderWidth = 4
  Caption = 'Panel2'
  TabOrder = 1
  <------->object DBGrid1:
TDBGrid
    Left = 6
    Top = 6
    Width = 486
    Height = 255
    Align = alClient
    BorderStyle = bsNone
    DataSource =
DmEmployee.EmployeeSource
    TabOrder = 0
    TitleFont.Charset =
DEFAULT_CHARSET
    TitleFont.Color = clBlack
    TitleFont.Height = -11
    TitleFont.Name = 'MS Sans
Serif'
    TitleFont.Style = []
  end
 end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

<------->object FrmQueryProc:
TFrmQueryProc
 Left = 279
 Top = 170
 Width = 381
 Height = 367
 Hint =
  'Explore the Get_Emp_Proj
procedure in the IBLOCAL alias to
see t' +
  'he query procedure'
 ActiveControl = Panel1
 Caption = 'Employee Project
Assignments'
 Color = clBtnFace
 Font.Charset =
DEFAULT_CHARSET
 Font.Color = clBlack
 Font.Height = -11
```

```
    Top = 8
    Width = 76
    Height = 25
    Hint = 'Rolls Back the current
transaction and starts a new one'
    Caption = '&Undo Edits'
    OnClick = BtnUndoEditsClick
  end
  <------->object BtnCommitEdits:
TSpeedButton
    Left = 261
    Top = 8
    Width = 76
    Height = 25
    Hint = 'Commits the current
transaction and starts a new one'
    Caption = '&Commit Edits'
    OnClick = BtnCommitEditsClick
  end
  <------->object DBNavigator:
TDBNavigator
    Left = 8
    Top = 8
    Width = 240
    Height = 25
    DataSource =
DmEmployee.EmployeeSource
    Ctl3D = False
    ParentCtl3D = False
    TabOrder = 0
  end
  <------->object BitBtn1: TBitBtn
    Left = 431
    Top = 8
    Width = 60
    Height = 25
    Hint = 'Exit and close this form'
    Caption = 'E&xit'
    TabOrder = 1
    Kind = bkClose
    Style = bsNew
  end
 end
 <------->object Panel2: TPanel
  Left = 0
  Top = 41
  Width = 498
  Height = 267
```

Caption = 'Panel2'
TabOrder = 1
<------->object DBGrid1: TDBGrid
Left = 6
Top = 6
Width = 361
Height = 125
Hint = 'Select an employee to execute the query procedure'
Align = alClient
BorderStyle = bsNone
DataSource = EmployeeSource
TabOrder = 0
TitleFont.Charset = DEFAULT_CHARSET
TitleFont.Color = clBlack
TitleFont.Height = -11
TitleFont.Name = 'MS Sans Serif'
TitleFont.Style = []
end
end
<------->object Panel3: TPanel
Left = 0
Top = 178
Width = 373
Height = 136
Align = alClient
BevelInner = bvLowered
BorderWidth = 4
TabOrder = 2
<------->object DBGrid2: TDBGrid
Left = 6
Top = 6
Width = 361
Height = 124
Hint = 'Use SQL Monitor to see the Get_Emp_Proj query procedure execute'
Align = alClient
BorderStyle = bsNone
DataSource = EmployeeProjectsSource
TabOrder = 0
TitleFont.Charset = DEFAULT_CHARSET

Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
ShowHint = True
OnHide = FormHide
OnShow = FormShow
PixelsPerInch = 96
TextHeight = 13
<------->object Panel1: TPanel
Left = 0
Top = 0
Width = 373
Height = 41
Align = alTop
TabOrder = 0
<------->object DBNavigator: TDBNavigator
Left = 8
Top = 8
Width = 240
Height = 25
DataSource = DmEmployee.EmployeeSource
Ctl3D = False
ParentCtl3D = False
TabOrder = 0
end
<------->object BitBtn1: TBitBtn
Left = 305
Top = 8
Width = 60
Height = 25
Hint = 'Exit and close this form'
Caption = 'E&xit'
TabOrder = 1
Kind = bkClose
Style = bsNew
end
end
<------->object Panel2: TPanel
Left = 0
Top = 41
Width = 373
Height = 137
Align = alTop
BevelInner = bvLowered
BorderWidth = 4

```
    Left = 168
    Top = 72
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
<------->object FrmLauncher: TFrmLauncher
  Left = 497
  Top = 119
  BorderIcons = [biSystemMenu, biMinimize]
  BorderStyle = bsSingle
  Caption = 'Client/Server Concepts'
  ClientHeight = 248
  ClientWidth = 240
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  ShowHint = True
  PixelsPerInch = 96
  TextHeight = 13
  <------->object BtnTrigg: TButton
    Left = 8
    Top = 48
    Width = 225
    Height = 34
    Hint = 'Shows a trigger in action'
    Caption = 'Salary Change &Trigger Demo'
    TabOrder = 1
    OnClick = BtnTriggClick
  end
  <------->object BtnViews: TButton
    Left = 8
    Top = 8
    Width = 225
    Height = 34
    Hint = 'Demonstrates that views are treated like tables'
    Caption = 'Show a &View in
      TitleFont.Color = clBlack
      TitleFont.Height = -11
      TitleFont.Name = 'MS Sans Serif'
      TitleFont.Style = []
    end
  end
  <------->object StatusBar1: TStatusBar
    Left = 0
    Top = 314
    Width = 373
    Height = 19
    Panels = <>
    SimplePanel = True
  end
  <------->object EmployeeProjectsQuery: TQuery
    DatabaseName = 'EmployeeDemoDB'
    SQL.Strings = (
      'Select * from Get_Emp_Proj( :EMP_NO )')
    Left = 168
    Top = 200
    ParamData = <
      item
        DataType = ftSmallint
        Name = 'EMP_NO'
        ParamType = ptUnknown
        Value = 2
      end>
  end
  <------->object EmployeeProjectsSource: TDataSource
    DataSet = EmployeeProjectsQuery
    Left = 168
    Top = 248
  end
  <------->object EmployeeSource: TDataSource
    DataSet = DmEmployee.EmployeeTable
    Enabled = False
    OnDataChange = EmployeeDataChange
```

```
  OnClick = BtnTransClick
 end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\\\\\\\\\\\\\\\

<------->object FrmExecProc: TFrmExecProc
 Left = 267
 Top = 122
 Width = 435
 Height = 316
 Hint =
  'Explore the ShipOrderProc in the DmEmployee data model to see wh' +
  'at shipping an order does'
 ActiveControl = Panel1
 Caption = 'Sales Review'
 Color = clBtnFace
 Font.Charset = DEFAULT_CHARSET
 Font.Color = clBlack
 Font.Height = -11
 Font.Name = 'MS Sans Serif'
 Font.Style = []
 OldCreateOrder = True
 Position = poScreenCenter
 ShowHint = True
 OnHide = FormHide
 OnShow = FormShow
 PixelsPerInch = 96
 TextHeight = 13
 <------->object Panel1: TPanel
  Left = 0
  Top = 0
  Width = 427
  Height = 41
  Align = alTop
  TabOrder = 0
  <------->object BtnShipOrder: TSpeedButton
   Left = 258
   Top = 8
   Width = 64
   Height = 25
   Hint = 'Mark current order as
```

```
action'
  TabOrder = 0
  OnClick = BtnViewsClick
 end
 <------->object BtnQrySP: TButton
  Left = 8
  Top = 88
  Width = 225
  Height = 34
  Hint = 'Shows a query procedure in action'
  Caption = '&Query Stored Procedure Demo'
  TabOrder = 2
  OnClick = BtnQrySPClick
 end
 <------->object BtnExecSP: TButton
  Left = 8
  Top = 128
  Width = 225
  Height = 34
  Hint = 'Shows an executable procedure in action'
  Caption = '&Executable Stored Procedure Demo'
  TabOrder = 3
  OnClick = BtnExecSPClick
 end
 <------->object BtnClose: TButton
  Left = 8
  Top = 208
  Width = 225
  Height = 34
  Hint = 'Exits this sample'
  Caption = '&Exit'
  TabOrder = 5
  OnClick = BtnCloseClick
 end
 <------->object BtnTrans: TButton
  Left = 8
  Top = 168
  Width = 225
  Height = 34
  Hint = 'Shows simple transaction handling'
  Caption = 'T&ransaction Editing Demo'
  TabOrder = 4
```

BorderStyle = bsNone
DataSource = DmEmployee.SalesSource
TabOrder = 0
TitleFont.Charset = DEFAULT_CHARSET
TitleFont.Color = clBlack
TitleFont.Height = -11
TitleFont.Name = 'MS Sans Serif'
TitleFont.Style = []
end
end
<------->object Panel3: TPanel
Left = 0
Top = 172
Width = 427
Height = 110
Hint = 'Customers are linked to sales in the data model'
Align = alBottom
BevelInner = bvLowered
BorderWidth = 4
Caption = 'Panel3'
TabOrder = 2
<------->object ScrollBox: TScrollBox
Left = 6
Top = 6
Width = 415
Height = 98
HorzScrollBar.Margin = 6
VertScrollBar.Margin = 6
Align = alClient
BorderStyle = bsNone
TabOrder = 0
<------->object Label1: TLabel
Left = 209
Top = 6
Width = 64
Height = 13
Alignment = taRightJustify
Caption = 'Customer No:'
FocusControl = EditCUST_NO
end
<------->object Label2: TLabel
Left = 1
Top = 7

shipped'
Caption = '&Ship Order'
Enabled = False
OnClick = BtnShipOrderClick
end
<------->object DBNavigator: TDBNavigator
Left = 8
Top = 8
Width = 240
Height = 25
DataSource = DmEmployee.SalesSource
Ctl3D = False
ParentCtl3D = False
TabOrder = 0
end
<------->object BitBtn1: TBitBtn
Left = 359
Top = 8
Width = 60
Height = 25
Hint = 'Exit and close this form'
Caption = 'E&xit'
TabOrder = 1
Kind = bkClose
Style = bsNew
end
end
<------->object Panel2: TPanel
Left = 0
Top = 41
Width = 427
Height = 131
Align = alClient
BevelInner = bvLowered
BorderWidth = 4
Caption = 'Panel2'
TabOrder = 1
<------->object DBGrid1: TDBGrid
Left = 6
Top = 6
Width = 415
Height = 119
Hint = 'Select an open order to ship the order'
Align = alClient

```
      Left = 51
      Top = 4
      Width = 125
      Height = 21
      DataField = 'CUSTOMER'
      DataSource =
DmEmployee.CustomerSource
      TabOrder = 1
    end
    <------->object
EditCONTACT_FIRST: TDBEdit
      Left = 51
      Top = 26
      Width = 75
      Height = 21
      DataField =
'CONTACT_FIRST'
      DataSource =
DmEmployee.CustomerSource
      TabOrder = 2
    end
    <------->object
EditCONTACT_LAST: TDBEdit
      Left = 129
      Top = 26
      Width = 100
      Height = 21
      DataField =
'CONTACT_LAST'
      DataSource =
DmEmployee.CustomerSource
      TabOrder = 3
    end
    <------->object EditPHONE_NO:
TDBEdit
      Left = 276
      Top = 26
      Width = 120
      Height = 21
      DataField = 'PHONE_NO'
      DataSource =
DmEmployee.CustomerSource
      TabOrder = 4
    end
    <------->object
EditADDRESS_LINE: TDBEdit
      Left = 51
      Top = 48
```

```
      Width = 47
      Height = 13
      Alignment = taRightJustify
      Caption = 'Company:'
      FocusControl =
EditCUSTOMER
    end
    <------->object Label3: TLabel
      Left = 8
      Top = 29
      Width = 40
      Height = 13
      Alignment = taRightJustify
      Caption = 'Contact:'
      FocusControl =
EditCONTACT_FIRST
    end
    <------->object Label6: TLabel
      Left = 7
      Top = 50
      Width = 41
      Height = 13
      Alignment = taRightJustify
      Caption = 'Address:'
      FocusControl =
EditADDRESS_LINE
    end
    <------->object Label4: TLabel
      Left = 239
      Top = 29
      Width = 34
      Height = 13
      Alignment = taRightJustify
      Caption = 'Phone:'
    end
    <------->object EditCUST_NO:
TDBEdit
      Left = 276
      Top = 3
      Width = 49
      Height = 21
      DataField = 'CUST_NO'
      DataSource =
DmEmployee.CustomerSource
      TabOrder = 0
    end
    <------->object EditCUSTOMER:
TDBEdit
```

Height = 21
DataField = 'COUNTRY'
DataSource = DmEmployee.CustomerSource
TabOrder = 9
end
<------->object EditPOSTAL_CODE: TDBEdit
Left = 336
Top = 70
Width = 60
Height = 21
DataField = 'POSTAL_CODE'
DataSource = DmEmployee.CustomerSource
TabOrder = 10
end
<------->object DBCheckBox1: TDBCheckBox
Left = 333
Top = 3
Width = 67
Height = 17
Caption = 'On Hold'
DataField = 'ON_HOLD'
DataSource = DmEmployee.CustomerSource
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
TabOrder = 11
ValueChecked = 'True;*;Yes'
ValueUnchecked = 'False; ;;NIL;No'
end
end
end
<------->object SalesSource: TDataSource
DataSet = DmEmployee.SalesTable
Enabled = False
OnDataChange = SalesSourceDataChange

Width = 171
Height = 21
DataField = 'ADDRESS_LINE1'
DataSource = DmEmployee.CustomerSource
TabOrder = 5
end
<------->object EditADDRESS_LINE2: TDBEdit
Left = 226
Top = 48
Width = 170
Height = 21
DataField = 'ADDRESS_LINE2'
DataSource = DmEmployee.CustomerSource
TabOrder = 6
end
<------->object EditCITY: TDBEdit
Left = 51
Top = 70
Width = 125
Height = 21
DataField = 'CITY'
DataSource = DmEmployee.CustomerSource
TabOrder = 7
end
<------->object EditSTATE_PROVINCE: TDBEdit
Left = 180
Top = 70
Width = 75
Height = 21
DataField = 'STATE_PROVINCE'
DataSource = DmEmployee.CustomerSource
TabOrder = 8
end
<------->object EditCOUNTRY: TDBEdit
Left = 258
Top = 70
Width = 75

Required = True
end
<------->object SalesTableSALES_REP: TSmallintField
DisplayLabel = 'Sales Rep'
FieldName = 'SALES_REP'
end
<------->object SalesTableORDER_STATUS: TStringField
DisplayLabel = 'Order Status'
FieldName = 'ORDER_STATUS'
Required = True
Size = 7
end
<------->object SalesTableORDER_DATE: TDateTimeField
DisplayLabel = 'Order Date'
FieldName = 'ORDER_DATE'
Required = True
end
<------->object SalesTableSHIP_DATE: TDateTimeField
DisplayLabel = 'Ship Date'
FieldName = 'SHIP_DATE'
end
<------->object SalesTableDATE_NEEDED: TDateTimeField
DisplayLabel = 'Date Needed'
FieldName = 'DATE_NEEDED'
end
<------->object SalesTablePAID: TStringField
DisplayLabel = 'Paid'
FieldName = 'PAID'
Size = 1
end
<------->object SalesTableQTY_ORDERED: TIntegerField
DisplayLabel = 'Quantity Ordered'
FieldName = 'QTY_ORDERED'
Required = True

Left = 200
Top = 80
end
End
\\ \\\
<------->object DmEmployee: TDmEmployee
OldCreateOrder = True
OnCreate = DmEmployeeCreate
Left = 126
Top = 130
Height = 188
Width = 414
<------->object EmployeeDatabase: TDatabase
AliasName = 'IBLOCAL'
Connected = True
DatabaseName = 'EmployeeDemoDB'
LoginPrompt = False
Params.Strings = (
'USER NAME=SYSDBA'
'PASSWORD=masterkey')
SessionName = 'Default'
Left = 40
Top = 8
end
<------->object SalesTable: TTable
DatabaseName = 'EmployeeDemoDB'
IndexFieldNames = 'PO_NUMBER'
TableName = 'SALES'
Left = 40
Top = 56
<------->object SalesTablePO_NUMBER: TStringField
DisplayLabel = 'PO Number'
FieldName = 'PO_NUMBER'
Required = True
Size = 8
end
<------->object SalesTableCUST_NO: TIntegerField
DisplayLabel = 'Customer No'
FieldName = 'CUST_NO'

Required = True
end
<------->object CustomerTableCUSTOMER: TStringField
DisplayLabel = 'Customer'
FieldName = 'CUSTOMER'
Required = True
Size = 25
end
<------->object CustomerTableCONTACT_FIRST: TStringField
DisplayLabel = 'Contact1'
FieldName = 'CONTACT_FIRST'
Size = 15
end
<------->object CustomerTableCONTACT_LAST: TStringField
DisplayLabel = 'Contact2'
FieldName = 'CONTACT_LAST'
end
<------->object CustomerTablePHONE_NO: TStringField
DisplayLabel = 'Phone No'
FieldName = 'PHONE_NO'
end
<------->object CustomerTableADDRESS_LINE1: TStringField
DisplayLabel = 'Address 1'
FieldName = 'ADDRESS_LINE1'
Size = 30
end
<------->object CustomerTableADDRESS_LINE2: TStringField
DisplayLabel = 'Address 2'
FieldName = 'ADDRESS_LINE2'
Size = 30
end
<------->object CustomerTableCITY: TStringField

end
<------->object SalesTableTOTAL_VALUE: TIntegerField
DisplayLabel = 'Total Value'
FieldName = 'TOTAL_VALUE'
Required = True
end
<------->object SalesTableDISCOUNT: TFloatField
DisplayLabel = 'Discount'
FieldName = 'DISCOUNT'
Required = True
end
<------->object SalesTableITEM_TYPE: TStringField
DisplayLabel = 'Item Type'
FieldName = 'ITEM_TYPE'
Size = 12
end
<------->object SalesTableAGED: TFloatField
DisplayLabel = 'Aged'
FieldName = 'AGED'
end
end
<------->object SalesSource: TDataSource
DataSet = SalesTable
Left = 40
Top = 104
end
<------->object CustomerTable: TTable
DatabaseName = 'EmployeeDemoDB'
IndexFieldNames = 'CUST_NO'
MasterFields = 'CUST_NO'
MasterSource = SalesSource
TableName = 'CUSTOMER'
Left = 128
Top = 56
<------->object CustomerTableCUST_NO: TIntegerField
DisplayLabel = 'Customer No'
FieldName = 'CUST_NO'

'EmployeeDemoDB'
IndexFieldNames = 'EMP_NO'
TableName = 'EMPLOYEE'
Left = 224
Top = 56
<------->object EmployeeTableEMP_NO: TSmallintField
DisplayLabel = 'Employee Number'
FieldName = 'EMP_NO'
Required = True
end
<------->object EmployeeTableFIRST_NAME: TStringField
DisplayLabel = 'First Name'
FieldName = 'FIRST_NAME'
Required = True
Size = 15
end
<------->object EmployeeTableLAST_NAME: TStringField
DisplayLabel = 'Last Name'
FieldName = 'LAST_NAME'
Required = True
end
<------->object EmployeeTablePHONE_EXT: TStringField
DisplayLabel = 'Phone Extension'
FieldName = 'PHONE_EXT'
Size = 4
end
<------->object EmployeeTableHIRE_DATE: TDateTimeField
DisplayLabel = 'Hire Date'
FieldName = 'HIRE_DATE'
Required = True
end
<------->object EmployeeTableDEPT_NO: TStringField
DisplayLabel = 'Dept No'
FieldName = 'DEPT_NO'

DisplayLabel = 'City'
FieldName = 'CITY'
Size = 25
end
<------->object CustomerTableSTATE_PROVINCE : TStringField
DisplayLabel = 'State or Province'
FieldName = 'STATE_PROVINCE'
Size = 15
end
<------->object CustomerTableCOUNTRY: TStringField
DisplayLabel = 'Country'
FieldName = 'COUNTRY'
Size = 15
end
<------->object CustomerTablePOSTAL_CODE: TStringField
DisplayLabel = 'Postal Code'
FieldName = 'POSTAL_CODE'
Size = 12
end
<------->object CustomerTableON_HOLD: TStringField
DisplayLabel = 'On Hold'
FieldName = 'ON_HOLD'
Size = 1
end
end
<------->object CustomerSource: TDataSource
DataSet = CustomerTable
Left = 128
Top = 104
end
<------->object EmployeeTable: TTable
AfterPost = EmployeeTableAfterPost
BeforeDelete = EmployeeTableBeforeDelete
DatabaseName =

TTable
DatabaseName = 'EmployeeDemoDB'
IndexFieldNames = 'EMP_NO'
MasterFields = 'EMP_NO'
MasterSource = EmployeeSource
TableName = 'SALARY_HISTORY'
Left = 328
Top = 56
<------->object SalaryHistoryTableEMPLOYEE: TStringField
DisplayLabel = 'Employee'
FieldKind = fkLookup
FieldName = 'EMPLOYEE'
LookupDataSet = EmployeeLookup
LookupKeyFields = 'EMP_NO'
LookupResultField = 'FULL_NAME'
KeyFields = 'EMP_NO'
Size = 37
Lookup = True
end
<------->object SalaryHistoryTableEMP_NO: TSmallintField
FieldName = 'EMP_NO'
Required = True
DisplayFormat = 'Employee Number'
end
<------->object SalaryHistoryTableCHANGE_DAT E: TDateTimeField
DisplayLabel = 'Change Date'
FieldName = 'CHANGE_DATE'
Required = True
end
<------->object SalaryHistoryTableUPDATER_ID: TStringField
DisplayLabel = 'Updater ID'
FieldName = 'UPDATER_ID'
Required = True
end
<------->object

Required = True
Size = 3
end
<------->object EmployeeTableJOB_CODE: TStringField
DisplayLabel = 'Job Code'
FieldName = 'JOB_CODE'
Required = True
Size = 5
end
<------->object EmployeeTableJOB_GRADE: TSmallintField
DisplayLabel = 'Job Grade'
FieldName = 'JOB_GRADE'
Required = True
end
<------->object EmployeeTableJOB_COUNTRY: TStringField
DisplayLabel = 'Job Country'
FieldName = 'JOB_COUNTRY'
Required = True
Size = 15
end
<------->object EmployeeTableSALARY: TFloatField
DisplayLabel = 'Salary'
FieldName = 'SALARY'
Required = True
end
<------->object EmployeeTableFULL_NAME: TStringField
DisplayLabel = 'Name'
FieldName = 'FULL_NAME'
Size = 37
end
end
<------->object EmployeeSource: TDataSource
DataSet = EmployeeTable
Left = 224
Top = 104
end
<------->object SalaryHistoryTable:

```
   Left = 224
   Top = 8
   ParamData = <
     item
       DataType = ftInteger
       Name = 'EMP_NUM'
       ParamType = ptInput
     end>
 end
 <------->object EmployeeLookup: TTable
   AfterPost = EmployeeTableAfterPost
   BeforeDelete = EmployeeTableBeforeDelete
   DatabaseName = 'EmployeeDemoDB'
   IndexFieldNames = 'EMP_NO'
   TableName = 'EMPLOYEE'
   Left = 328
   Top = 8
   <------->object SmallintField1: TSmallintField
     DisplayLabel = 'Employee Number'
     FieldName = 'EMP_NO'
     Required = True
   end
   <------->object StringField1: TStringField
     DisplayLabel = 'First Name'
     FieldName = 'FIRST_NAME'
     Required = True
     Size = 15
   end
   <------->object StringField2: TStringField
     DisplayLabel = 'Last Name'
     FieldName = 'LAST_NAME'
     Required = True
   end
   <------->object StringField3: TStringField
     DisplayLabel = 'Phone Extension'
     FieldName = 'PHONE_EXT'
     Size = 4
   end
```

```
SalaryHistoryTableOLD_SALARY: TFloatField
     DisplayLabel = 'Old Salary'
     FieldName = 'OLD_SALARY'
     Required = True
   end
   <------->object SalaryHistoryTablePERCENT_CHANGE: TFloatField
     DisplayLabel = '% Change'
     FieldName = 'PERCENT_CHANGE'
     Required = True
   end
   <------->object SalaryHistoryTableNEW_SALARY: TFloatField
     DisplayLabel = 'New Salary'
     FieldName = 'NEW_SALARY'
   end
 end
 <------->object SalaryHistorySource: TDataSource
   DataSet = SalaryHistoryTable
   Left = 328
   Top = 104
 end
 <------->object ShipOrderProc: TStoredProc
   DatabaseName = 'EmployeeDemoDB'
   StoredProcName = 'SHIP_ORDER'
   Left = 128
   Top = 8
   ParamData = <
     item
       DataType = ftString
       Name = 'PO_NUM'
       ParamType = ptInput
     end>
 end
 <------->object DeleteEmployeeProc: TStoredProc
   DatabaseName = 'EmployeeDemoDB'
   StoredProcName = 'DELETE_EMPLOYEE'
```

\\\

<------->object DateTimeField1: TDateTimeField
DisplayLabel = 'Hire Date'
FieldName = 'HIRE_DATE'
Required = True
end
<------->object StringField4: TStringField
DisplayLabel = 'Dept No'
FieldName = 'DEPT_NO'
Required = True
Size = 3
end
<------->object StringField5: TStringField
DisplayLabel = 'Job Code'
FieldName = 'JOB_CODE'
Required = True
Size = 5
end
<------->object SmallintField2: TSmallintField
DisplayLabel = 'Job Grade'
FieldName = 'JOB_GRADE'
Required = True
end
<------->object StringField6: TStringField
DisplayLabel = 'Job Country'
FieldName = 'JOB_COUNTRY'
Required = True
Size = 15
end
<------->object FloatField1: TFloatField
DisplayLabel = 'Salary'
FieldName = 'SALARY'
Required = True
end
<------->object StringField7: TStringField
DisplayLabel = 'Name'
FieldName = 'FULL_NAME'
Size = 37
end
end
End

(Test program for ADO Components)
موقعه : دلفي ٧ (النسخة الأصلية)
الملف النصي والتنفيذي

Procedures159

unit AdoMain;
{ Test program for ADO Components }
interface
uses
Variants, Windows, Sysutils, Forms, IniFiles, ImgList, Controls, Classes,
ActnList, Menus, Dialogs, ComCtrls, ComObj, ToolWin, DB, ADOInt,
Grids, DBGrids, Provider, ADODB, DBClient, DBCtrls, ExtCtrls,
StdCtrls, Buttons, SqlEdit;
type
TADODBTest = class(TForm)
Connection: TADOConnection;
MasterTable: TADOTable;
DetailTable: TADOTable;
MasterQuery: TADOQuery;
DetailQuery: TADOQuery;
MasterProc: TADOStoredProc;
ADODataSet: TADODataSet;
Provider: TDataSetProvider;
MasterClientData: TClientDataSet;
MasterDataSource: TDataSource;
DetailDataSource: TDataSource;
DetailGrid: TDBGrid;
MasterGrid: TDBGrid;
DBMemo1: TDBMemo;
DBImage1: TDBImage;
{ Actions }
ActionList1: TActionList;
SaveToFile: TAction;

البرنـامج الشامل لتعليمات ADO AdoTest٥
إعداد :
علاء الدين محمد اللباد
٠٩٤٤٥٧٥٣٧١
اسم البرنـامج :

FindNext: TButton;
Filtered: TCheckBox;
IndexFields: TEdit;
PrevQuery1: TSpeedButton;
Events: TListBox;
DescFields: TEdit;
CaseInsFields: TEdit;
MasterTableName: TComboBox;
DetailTableName: TComboBox;
MasterSQL: TMemo;
DetailSQL: TMemo;
GridSplitter: TSplitter;
ClearEventsButton: TToolButton;
LocateEdit: TEdit;
LocateField: TComboBox;
locPartialKey: TCheckBox;
LocateNull: TCheckBox;
UseClientCursorItem: TMenuItem;
UseadCmdTableDirect1: TMenuItem;
CursorTypeItem: TMenuItem;
CurTypeKeyset: TMenuItem;
Dynamic1: TMenuItem;
CurTypeUnspecified: TMenuItem;
CurTypeForwardOnly: TMenuItem;
CurTypeStatic: TMenuItem;
DBEditScroller: TScrollBox;
LockTypeItem: TMenuItem;
LckTypeUnspecified: TMenuItem;
LckTypeReadOnly: TMenuItem;
LckTypePessimistic: TMenuItem;
LckTypeOptimistic: TMenuItem;
LckTypeBatchOptimistic: TMenuItem;
ReadOnlyLabel: TLabel;
ConnectionString: TComboBox;
EditConnStr: TSpeedButton;
Label1: TLabel;
Label2: TLabel;
ProcedureNames: TGroupBox;
TableNames: TGroupBox;
QueryStrings: TGroupBox;
MasterProcName: TComboBox;
Splitter1: TSplitter;

OpenQuery: TAction;
OpenTable: TAction;
BatchUpdate: TAction;
ExitApplication: TAction;
CloseActiveDataSet: TAction;
LoadFromFile: TAction;
CancelBatch: TAction;
ExecuteCommand: TAction;
StreamFormOut: TAction;
StreamFormIn: TAction;
ClearField: TAction;
ViewEvents: TAction;
PrevQuery: TAction;
NextQuery: TAction;
RefreshData: TAction;
ClearEventLog: TAction;
DisplayDetails: TAction;
HelpAbout: TAction;
UseClientCursor: TAction;
UseTableDirect: TAction;
UseShapeProvider: TAction;
AsyncConnect: TAction;
AsyncExecute: TAction;
AsyncFetch: TAction;
OpenProcedure: TAction;
MainMenu1: TMainMenu;
FileReopen: TMenuItem;
FileMenu: TMenuItem;
PopupMenu1: TPopupMenu;
ToolBar1: TToolBar;
ImageList1: TImageList;
OpenDialog: TOpenDialog;
SaveDialog: TSaveDialog;
StatusBar: TStatusBar;
AreaSelector: TPageControl;
DataPanel: TPanel;
FilterPage: TTabSheet;
LocatePage: TTabSheet;
IndexPage: TTabSheet;
FieldsPage: TTabSheet;
SourcePage: TTabSheet;
IndexList: TListBox;
NavigatorPanel: TPanel;
BlobCtrlPanel: TPanel;
GridPanel: TPanel;
DBNavigator1: TDBNavigator;
Filter: TEdit;
FindFirst: TButton;

TButton;
ParameterType: TComboBox;
ToolButton3: TToolButton;
MidasApplyUpdatesButton: TToolButton;
MidasApplyUpdates: TAction;
ADOButton: TRadioButton;
MidasButton: TRadioButton;
MidasCancelUpdates: TAction;
MidasCancelButton: TToolButton;
ApplyUpdatesMidas1: TMenuItem;
CancelUpdatesMidas1: TMenuItem;
N6: TMenuItem;
SQLParams: TRadioButton;
ProcParams: TRadioButton;
TestButton: TButton;
FieldSchemaGrid: TDBGrid;
FieldSchemaSource: TDataSource;
FieldSchema: TADODataSet;
FieldSchemaCOLUMN_NAME: TWideStringField;
FieldSchemaDATA_TYPE: TWordField;
FieldSchemaNUMERIC_PRECISION: TWordField;
FieldSchemaCHARACTER_MAXIMUM_LENGTH: TIntegerField;
FieldSchemaNUMERIC_SCALE: TSmallintField;
EnableBCD: TAction;
EnableBCD1: TMenuItem;
DisconnectDataSet: TAction;
DisconnectDataSet1: TMenuItem;
DetailClientData: TClientDataSet;
FilterGroupBox: TRadioGroup;
BlobAsImage: TAction;
BlobfieldasImage1: TMenuItem;
LoadBlobFromFile: TAction;
LoadBlobfromfile1: TMenuItem;
IndexOptions: TGroupBox;
idxCaseInsensitive: TCheckBox;
idxDescending: TCheckBox;

DetailMasterSource: TDataSource;
DetailQuerySource: TDataSource;
CloseConnection: TAction;
Disconnect1: TMenuItem;
BatchUpdates1: TMenuItem;
CancelBatch1: TMenuItem;
BatchUpdateButton: TToolButton;
CancelBatchButton: TToolButton;
AsyncConnect1: TMenuItem;
AsyncExecute1: TMenuItem;
AsyncFetch1: TMenuItem;
ADOCommand: TADOCommand;
ProgressBar: TProgressBar;
MaxRecords: TAction;
MaxRecords1: TMenuItem;
DetailProcName: TComboBox;
DetailProc: TADOStoredProc;
ToolButton1: TToolButton;
OpenProcedure1: TMenuItem;
DetailProcSource: TDataSource;
EditCommandText: TSpeedButton;
ParamPage: TTabSheet;
ParameterList: TListBox;
ParameterName: TEdit;
ParameterValue: TEdit;
ParameterSize: TEdit;
ParameterNameLabel: TLabel;
ParameterScale: TEdit;
ParameterPrecision: TEdit;
PTypeLabel: TLabel;
PValueLabel: TLabel;
PSizeLabel: TLabel;
PScaleLabel: TLabel;
PPrecisionLabel: TLabel;
ParameterDirectionGroup: TRadioGroup;
ParamAttributes: TGroupBox;
PANullableCheckBox: TCheckBox;
PASignedCheckBox: TCheckBox;
PALongCheckBox: TCheckBox;
AddParameterButton: TButton;
RefreshParametersButton:

StreamFormInClick(Sender: TObject);
<------> procedure LoadFromFileExecute(Sender: TObject);
<------> procedure SaveToFileExecute(Sender: TObject);
<------> procedure EditActionsUpdate(Sender: TObject);
<------> procedure FieldsPageShow(Sender: TObject);
<------> procedure OpenQueryExecute(Sender: TObject);
<------> procedure ExecSQLExecute(Sender: TObject);
<------> procedure OpenTableExecute(Sender: TObject);
<------> procedure BatchUpdateExecute(Sender: TObject);
<------> procedure MasterTableNameDropDown(Sender: TObject);
<------> procedure ConnectionStringClick(Sender: TObject);
<------> procedure ConnectionStringKeyPress(Sender: TObject; var Key: Char);
<------> procedure FilteredClick(Sender: TObject);
<------> procedure FilterPageShow(Sender: TObject);
<------> procedure IndexPageShow(Sender: TObject);
<------> procedure ExitApplicationExecute(Sender: TObject);
<------> procedure CloseActiveDataSetExecute(Sender: TObject);
<------> procedure FileActionsUpdate(Sender: TObject);
<------> procedure

idxPrimary: TCheckBox;
idxUnique: TCheckBox;
<------> procedure FilterKeyPress(Sender: TObject; var Key: Char);
<------> procedure FormCreate(Sender: TObject);
<------> procedure FormDestroy(Sender: TObject);
<------> procedure MasterSQLKeyPress(Sender: TObject; var Key: Char);
<------> procedure IndexListClick(Sender: TObject);
<------> procedure GridTitleClick(Column: TColumn);
<------> procedure LocateButtonClick(Sender: TObject);
<------> procedure FindFirstClick(Sender: TObject);
<------> procedure FilterExit(Sender: TObject);
<------> procedure DataSourceDataChange(Sender: TObject; Field: TField);
<------> procedure DataSetAfterOpen(DataSet: TDataSet);
<------> procedure LocateFieldDropDown(Sender: TObject);
<------> procedure FindNextClick(Sender: TObject);
<------> procedure MasterTableNameClick(Sender: TObject);
<------> procedure PopupMenu1Popup(Sender: TObject);
<------> procedure FieldSelect(Sender: TObject);
<------> procedure GridColEnter(Sender: TObject);
<------> procedure StreamFormOutClick(Sender: TObject);
<------> procedure

DataSetNewRecord(DataSet: TDataSet);
<------> procedure DataSetAfterPost(DataSet: TDataSet);
<------> procedure DataSetAfterInsert(DataSet: TDataSet);
<------> procedure DataSetAfterEdit(DataSet: TDataSet);
<------> procedure DataSetAfterDelete(DataSet: TDataSet);
<------> procedure DataSetAfterCancel(DataSet: TDataSet);
<------> procedure MasterQueryAfterOpen(DataSet: TDataSet);
<------> procedure MasterQueryBeforeClose(DataSet: TDataSet);
<------> procedure CancelBatchExecute(Sender: TObject);
<------> procedure ClearFieldExecute(Sender: TObject);
<------> procedure ViewEventsExecute(Sender: TObject);
<------> procedure DisplayDetailsExecute(Sender: TObject);
<------> procedure DataSourceStateChange(Sender: TObject);
<------> procedure DataSourceUpdateData(Sender: TObject);
<------> procedure RefreshDataExecute(Sender: TObject);
<------> procedure ClearEventLogExecute(Sender: TObject);
<------> procedure ClearEventLogUpdate(Sender:

MasterTableNameKeyPress(Sender: TObject; var Key: Char);
<------> procedure DetailTableNameClick(Sender: TObject);
<------> procedure MasterTableAfterOpen(DataSet: TDataSet);
<------> procedure MasterTableBeforeClose(DataSet: TDataSet);
<------> procedure GridSetFocus(Sender: TObject);
<------> procedure LocatePageShow(Sender: TObject);
<------> procedure LocateNullClick(Sender: TObject);
<------> procedure DataSetAfterScroll(DataSet: TDataSet);
<------> procedure DataSetBeforeCancel(DataSet: TDataSet);
<------> procedure DataSetBeforeClose(DataSet: TDataSet);
<------> procedure DataSetBeforeDelete(DataSet: TDataSet);
<------> procedure DataSetBeforeEdit(DataSet: TDataSet);
<------> procedure DataSetBeforeInsert(DataSet: TDataSet);
<------> procedure DataSetBeforePost(DataSet: TDataSet);
<------> procedure DataSetBeforeScroll(DataSet: TDataSet);
<------> procedure DataSetCalcFields(DataSet: TDataSet);
<------> procedure DataSetError(DataSet: TDataSet; E: EDatabaseError;
var Action: TDataAction);
<------> procedure

<------> procedure DetailQueryBeforeOpen(DataSet: TDataSet);
<------> procedure MasterProcBeforeOpen(DataSet: TDataSet);
<------> procedure UseShapeProviderExecute(Sender: TObject);
<------> procedure OnFilterRecord(DataSet: TDataSet; var Accept: Boolean);
<------> procedure BinaryGetText(Sender: TField; var Text: string;
DisplayText: Boolean);
<------> procedure BinarySetText(Sender: TField; const Text: string);
<------> procedure ConnectionBeginTransComplete(Connection: TADOConnection;
TransactionLevel: Integer; const Error: Error;
var EventStatus: TEventStatus);
<------> procedure ConnectionCommitTransComplete(Connection: TADOConnection;
const Error: Error; var EventStatus: TEventStatus);
<------> procedure ConnectionConnectComplete(Connection: TADOConnection;
const Error: Error; var EventStatus: TEventStatus);
<------> procedure ConnectionDisconnect(Connection: TADOConnection;
var EventStatus: TEventStatus);
<------> procedure ConnectionExecuteComplete(Connection: TADOConnection;
RecordsAffected: Integer; const Error: Error;
var EventStatus: TEventStatus; const Command: _Command;
const Recordset: _Recordset);
<------> procedure ConnectionInfoMessage(Connection

TObject);
<------> procedure HelpAboutExecute(Sender: TObject);
<------> procedure DataSetAfterClose(DataSet: TDataSet);
<------> procedure FileMenuClick(Sender: TObject);
<------> procedure ClosedFileClick(Sender: TObject);
<------> procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
<------> procedure PrevQueryExecute(Sender: TObject);
<------> procedure PrevQueryUpdate(Sender: TObject);
<------> procedure NextQueryExecute(Sender: TObject);
<------> procedure MasterTableAfterScroll(DataSet: TDataSet);
<------> procedure MasterTableBeforeScroll(DataSet: TDataSet);
<------> procedure RadioItemClick(Sender: TObject);
<------> procedure DataSetBeforeOpen(DataSet: TDataSet);
<------> procedure BooleanActionExecute(Sender: TObject);
<------> procedure EditConnStrClick(Sender: TObject);
<------> procedure MasterTableBeforeOpen(DataSet: TDataSet);
<------> procedure DetailTableBeforeOpen(DataSet: TDataSet);
<------> procedure MasterQueryBeforeOpen(DataSet: TDataSet);

TObject);
<------> procedure ProcNameDropDown(Sender: TObject);
<------> procedure MasterProcNameKeyPress(Sender: TObject; var Key: Char);
<------> procedure MasterProcNameClick(Sender: TObject);
<------> procedure DetailProcNameClick(Sender: TObject);
<------> procedure OpenProcedureExecute(Sender: TObject);
<------> procedure DetailProcBeforeOpen(DataSet: TDataSet);
<------> procedure MasterProcAfterOpen(DataSet: TDataSet);
<------> procedure EditCommandTextClick(Sender: TObject);
<------> procedure ParamPageShow(Sender: TObject);
<------> procedure RefreshParametersButtonClick(Sender: TObject);
<------> procedure ParameterListClick(Sender: TObject);
<------> procedure AddParameterButtonClick(Sender: TObject);
<------> procedure ParameterDataChange(Sender: TObject);
<------> procedure MasterSQLChange(Sender: TObject);
<------> procedure MidasApplyUpdatesExecute(Sender : TObject);
<------> procedure ADOButtonClick(Sender: TObject);
<------> procedure MidasButtonClick(Sender:

: TADOConnection;
const Error: Error; var EventStatus: TEventStatus);
<------> procedure ConnectionRollbackTransComplete(Connection: TADOConnection;
const Error: Error; var EventStatus: TEventStatus);
<------> procedure ConnectionWillConnect(Connection : TADOConnection;
var ConnectionString, UserID, Password: WideString;
var ConnectOptions: TConnectOption; var EventStatus: TEventStatus);
<------> procedure ConnectionWillExecute(Connection: TADOConnection;
var CommandText: WideString; var CursorType: TCursorType;
var LockType: TADOLockType; var CommandType: TCommandType;
var ExecuteOptions: TExecuteOptions; var EventStatus: TEventStatus;
const Command: _Command; const Recordset: _Recordset);
<------> procedure CloseConnectionExecute(Sender: TObject);
<------> procedure DataSetFetchComplete(DataSet: TCustomADODataSet;
const Error: Error; var EventStatus: TEventStatus);
<------> procedure ExceptionHandler(Sender: TObject; E: Exception);
<------> procedure ConnectionLogin(Sender: TObject; Username, Password: String);
<------> procedure MasterGridColumnMoved(Sender: TObject; FromIndex,
ToIndex: Integer);
<------> procedure MaxRecordsExecute(Sender:

TCustomADODataSet;
FActiveDataSource: TDataSource;
FStatusMsg: string;
FClosedTables: TStringList;
FMasterQueries: TStringList;
FDetailQueries: TStringList;
FQueryIndex: Integer;
FModifiedParameter: Integer;
FMovingColumn: Boolean;
FParamSource: TParameters;
FLastDataFile: String;
FLastFormFile: String;
function GetConfigFile: TIniFile;
<------> **procedure RefreshIndexNames;**
<------> **procedure SetActiveDataSet(Value: TDataSet);**
<------> **procedure SetEventsVisible(Visible: Boolean);**
<------> **procedure SetQueryText;**
<------> **procedure SetStatusMsg(const Msg: string);**
<------> **procedure ShowHeapStatus(Sender: TObject; var Done: Boolean);**
<------> **procedure UpdateReOpenMenu;**
<------> **procedure OnHint(Sender: TObject);**
<------> **procedure ClearProgressBar;**
<------> **procedure ShowProgressBar(const Msg: string);**
<------> **procedure ProcessQuery(SelectQuery: Boolean);**
<------> **procedure WriteParameterData;**
<------> **procedure UpdateParameterList;**
<------> **procedure ShowIndexParams;**
<------> **procedure SetRecordSetEvents(Hook: Boolean; DataSet: TCustomADODataSet);**
function GetActiveDataSet:

TObject);
<------> **procedure MasterClientDataReconcileError(DataSet: TCustomClientDataSet;**
E: EReconcileError; UpdateKind: TUpdateKind;
var Action: TReconcileAction);
<------> **procedure MidasCancelUpdatesExecute(Sender: TObject);**
<------> **procedure ParameterSourceClick(Sender: TObject);**
<------> **procedure FieldSchemaDATA_TYPEGetText(Sender: TField; var Text: String;**
DisplayText: Boolean);
<------> **procedure DisconnectDataSetExecute(Sender: TObject);**
<------> **procedure FieldValidate(Sender: TField);**
<------> **procedure TestButtonClick(Sender: TObject);**
<------> **procedure DataSetFetchProgress(DataSet: TCustomADODataSet; Progress,**
MaxProgress: Integer; var EventStatus: TEventStatus);
<------> **procedure FilterGroupBoxClick(Sender: TObject);**
<------> **procedure BlobAsImageUpdate(Sender: TObject);**
<------> **procedure BlobAsImageExecute(Sender: TObject);**
<------> **procedure LoadBlobFromFileExecute(Sender: TObject);**
<------> **procedure ToolBar1Click(Sender: TObject);**
private
FConfig: TIniFile;
FMaxErrors: Integer;
FPacketRecs: Integer;
FActiveDataSet: TDataSet;
FADOSource:

```
begin
  F := CreateMessageDialog('',
mtInformation, [mbCancel]);
  F.Height := Screen.Height div 2;
  F.Width := Screen.Width div 2;
  Button := F.Components[2] as
TButton;
  Button.Top := F.ClientHeight -
Button.Height - 5;
  Button.Left := (F.ClientWidth -
Button.Width) div 2;
  F.Caption := 'Properties';
  with TMemo.Create(F) do
  begin
    SetBounds(5, 5, F.ClientWidth-10,
F.ClientHeight - 40);
    Parent := F;
    for I := 0 to Props.Count - 1 do
      with Props[I] do
        Lines.Add(Format('%-30s: %s',
[Name, VarToStr(Value)]));
  end;
  F.ShowModal;
end;

{$R *.dfm}

procedure
TADODBTest.FormCreate(Sender:
TObject);

  procedure SetupControls;
  var
    I: Integer;
  begin
    for I := 0 to
StatusBar.Panels.Count - 1 do
      StatusBar.Panels[I].Text := '';
    ProgressBar.Parent := StatusBar;
    ProgressBar.SetBounds(0, 2,
StatusBar.Panels[0].Width,
StatusBar.Height - 2);
    { Set these dynamically since the
form may have been scaled }
    DataPanel.Constraints.MinWidth
:= DataPanel.Width;

AreaSelector.Constraints.MinWidth
```

```
TDataSet;
  public
<------>    procedure
BindControls(DataSet: TDataSet);
<------>    procedure
CheckConnection(CloseFirst:
Boolean = False);
<------>    procedure
OpenDataSet(Source:
TCustomADODataSet);
<------>    procedure
StreamSettings(Write: Boolean);
<------>    procedure LogEvent(const
EventStr: string; Component:
TComponent = nil);
<------>    procedure
RefreshParameters(Parameters:
TParameters);
    property StatusMsg: string read
FStatusMsg write SetStatusMsg;
    property ActiveDataSet: TDataSet
read GetActiveDataSet write
SetActiveDataSet;
    property ActiveDataSource:
TDataSource read
FActiveDataSource write
FActiveDataSource;
    property ADOSource:
TCustomADODataSet read
FADOSource write FADOSource;
    property ConfigFile: TIniFile read
GetConfigFile;
  end;

var
  ADODBTest: TADODBTest;

implementation

uses
  OLEDB, DBLogDlg, AdoConEd,
RecError;

procedure ShowProperties(Props:
Properties);
var
  I: Integer;
  F: TForm;
  Button: TButton;
```

```
  Application.Terminate;
end;

procedure
TADODBTest.HelpAboutExecute(Se
nder: TObject);
begin

ShowMessage(Caption+#13#10+'Co
pyright (c) 1999-2002 Borland
Corporation');
end;

procedure
TADODBTest.OnHint(Sender:
TObject);
begin
  if
FindVCLWindow(Mouse.CursorPos
) <> ConnectionString then
    ConnectionString.Hint :=
ConnectionString.Text;
  StatusMsg := Application.Hint;
end;

procedure
TADODBTest.ExceptionHandler(Se
nder: TObject; E: Exception);
begin
  ClearProgressBar;

SysUtils.ShowException(ExceptObje
ct, ExceptAddr);
end;

{ View Options }

procedure
TADODBTest.SetEventsVisible(Visi
ble: Boolean);
var
  EventsWidth: Integer;
begin
  Constraints.MinWidth := 0;
  if Events.Visible <> Visible then
  begin
    DataPanel.Anchors :=
DataPanel.Anchors - [akRight];
    AreaSelector.Anchors :=
:= AreaSelector.Width;
    Constraints.MinHeight := Height -
(DataPanel.Height -
DataPanel.Constraints.MinHeight);

SetEventsVisible(ViewEvents.Check
ed);
  end;

begin
  FMaxErrors := -1;
  FPacketRecs := -1;
  FModifiedParameter := -1;
  ActiveDataSource :=
MasterDataSource;

SetCurrentDirectory(PChar(Extract
FilePath(ParamStr(0))));
  Application.OnIdle :=
ShowHeapStatus;
  Application.OnHint := OnHint;
  FClosedTables :=
TStringList.Create;
  FMasterQueries :=
TStringList.Create;
  FDetailQueries :=
TStringList.Create;
  StreamSettings(False);
  SetupControls;
  ParameterSourceClick(Self);
end;

procedure
TADODBTest.FormDestroy(Sender:
TObject);
begin
  if Assigned(FConfig) then
    StreamSettings(True);
  FConfig.Free;
  FDetailQueries.Free;
  FMasterQueries.Free;
  FClosedTables.Free;
end;

procedure
TADODBTest.ExitApplicationExecu
te(Sender: TObject);
begin
```

```
begin
  if FConfig = nil then
    FConfig :=
TIniFile.Create(ChangeFileExt(ParamStr(0), '.INI'));
  Result := FConfig;
end;

procedure
TADODBTest.StreamSettings(Write: Boolean);

  procedure WriteStr(const
OptName: string; Value: Variant);
  begin
    FConfig.WriteString('Settings',
OptName, Value);
  end;

  procedure WriteBool(const
OptName: string; Value: Boolean);
  begin
    FConfig.WriteBool('Settings',
OptName, Value);
  end;

  procedure WriteStrings(const
SectName: string; Values: TStrings);
  var
    I: Integer;
  begin
    FConfig.EraseSection(SectName);
    for I := 0 to Values.Count - 1 do
      FConfig.WriteString(SectName,
IntToStr(I), Values[I]);
  end;

  function ReadStr(const OptName:
string): Variant;
  begin
    Result :=
FConfig.ReadString('Settings',
OptName, '');
  end;

  function ReadBool(const OptName:
string): Boolean;
  begin
```

```
AreaSelector.Anchors  - [akRight];
    try
      EventsWidth := Events.Width +
5;
      Events.Visible := Visible;
      if not Visible then
        EventsWidth := -EventsWidth;
      ClientWidth := ClientWidth +
EventsWidth;
    finally
      DataPanel.Anchors :=
DataPanel.Anchors + [akRight];
      AreaSelector.Anchors :=
AreaSelector.Anchors  + [akRight];
    end;
  end;
  if Visible then
    Constraints.MinWidth :=
DataPanel.Constraints.MinWidth +
Events.Width + 22 else
    Constraints.MinWidth :=
DataPanel.Constraints.MinWidth +
18;
end;

procedure
TADODBTest.ViewEventsExecute(Sender: TObject);
begin
  ViewEvents.Checked := not
ViewEvents.Checked;

SetEventsVisible(ViewEvents.Checked);
end;

procedure
TADODBTest.DisplayDetailsExecute(Sender: TObject);
begin
  DisplayDetails.Checked := not
DisplayDetails.Checked;
end;

{ Settings }

function
TADODBTest.GetConfigFile:
TIniFile;
```

with TEdit(Components[I]) do
WriteStr(Name, Text)
else if Components[I] is TComboBox then
with TDBComboBox(Components[I]) do
WriteStr(Name, Text)
else if Components[I] is TCheckBox then
with TCheckBox(Components[I]) do
WriteBool(Name, Checked)
else if Components[I] is TRadioButton then
with TRadioButton(Components[I]) do
WriteBool(Name, Checked)
else if Components[I] is TAction then
with TAction(Components[I]) do
WriteBool(Name, Checked)
else if Components[I] is TPageControl then
with TPageControl(Components[I]) do
WriteStr(Name, ActivePage.Caption)
else if Components[I] is TMenuItem then
with TMenuItem(Components[I]) do
for J := 0 to Count-1 do
if Items[J].Checked then
begin
WriteStr(Name, J);
System.Break;
end;
end
else
begin
for I := Low(Components) to High(Components) do
if Components[I] is TCustomEdit then
with TEdit(Components[I]) do
Text := ReadStr(Name)
else if Components[I] is

Result := FConfig.ReadBool('Settings', OptName, False);
end;
procedure ReadStrings(const SectName: string; Values: TStrings);
var
I: Integer;
S: string;
begin
for I := 0 to 99 do
begin
S := FConfig.ReadString(SectName, IntToStr(I), '');
if S = '' then Break;
Values.Add(S);
end;
end;
function FindPage(const PageName: string): TTabSheet;
var
I: Integer;
begin
for I := AreaSelector.PageCount - 1 downto 0 do
begin
Result := AreaSelector.Pages[I];
if Result.Caption = PageName then Exit;
end;
Result := SourcePage;
end;
procedure ProcessComponents(Components: array of TComponent);
var
I,J: Integer;
begin
if Write then
begin
for I := Low(Components) to High(Components) do
if Components[I] is TCustomEdit then

```
MasterProcName, DetailProcName, MasterSQL, DetailSQL, ViewEvents, DisplayDetails,
   UseClientCursor, UseTableDirect, UseShapeProvider, CursorTypeItem,
   LockTypeItem, AsyncConnect, AsyncExecute, AsyncFetch, MidasButton,
   ProcParams, EnableBCD]);
  if Write then
  begin
   WriteStrings('ConnectionStrings', ConnectionString.Items);
   WriteStrings('ClosedTables', FClosedTables);
   WriteStrings('MasterQueries', FMasterQueries);
   WriteStrings('DetailQueries', FDetailQueries);
   FConfig.UpdateFile;
  end else
  begin
   ReadStrings('ConnectionStrings', ConnectionString.Items);
   ReadStrings('ClosedTables', FClosedTables);
   ReadStrings('MasterQueries', FMasterQueries);
   ReadStrings('DetailQueries', FDetailQueries);
  end;
end;

procedure TADODBTest.RadioItemClick(Sender: TObject);
begin
  (Sender as TMenuItem).Checked := True;
end;

procedure TADODBTest.BooleanActionExecute(Sender: TObject);
begin
  TAction(Sender).Checked := not TAction(Sender).Checked;
end;
```

```
TComboBox then
        with TComboBox(Components[I]) do
          Text := ReadStr(Name)
      else if Components[I] is TCheckBox then
        with TCheckBox(Components[I]) do
          Checked := ReadBool(Name)
      else if Components[I] is TRadioButton then
        with TRadioButton(Components[I]) do
          Checked := ReadBool(Name)
      else if Components[I] is TAction then
        with TAction(Components[I]) do
          Checked := ReadBool(Name)
      else if Components[I] is TPageControl then
        with TPageControl(Components[I]) do
          ActivePage := FindPage(ReadStr(Name))
      else if Components[I] is TMenuItem then
        with TMenuItem(Components[I]) do
          Items[ReadStr(Name)].Checked := True;
    end;
  end;

begin
  GetConfigFile;
  if not Write and (ReadStr('AreaSelector') = '') then
  begin
    ConnectionString.Text := 'FILE NAME=' + DataLinkDir + '\DBDEMOS.UDL';
    Exit;
  end;
  ProcessComponents([AreaSelector, ConnectionString, MasterTableName,
    DetailTableName,
```

```
  StatusBar.Panels[3].Text := Msg+'...';
  while ProgressBar.Visible do
  begin
    ProgressBar.StepIt;
    Application.ProcessMessages;
    Sleep(ProgressBar.Position);
  end;
end;

procedure TADODBTest.ClearProgressBar;
begin
  ProgressBar.Hide;
  ProgressBar.Position := 0;
  StatusBar.Panels[3].Text := '';
end;

procedure TADODBTest.DataSourceDataChange(Sender: TObject;
  Field: TField);
const
  StatusStrs: array[TUpdateStatus] of string = ('Unmodified',
    'Modified', 'Inserted', 'Deleted');
begin
  if (Sender = ActiveDataSource) and Assigned(ActiveDataSource.DataSet) and ActiveDataSource.DataSet.IsSequenced then
  begin
    with ActiveDataSource.DataSet do
    begin
      if IsEmpty then
      begin
        StatusBar.Panels[1].Text := '';
        StatusBar.Panels[3].Text := '(empty)';
      end else
      begin
        StatusBar.Panels[1].Text := StatusStrs[UpdateStatus];
        if (State = dsBrowse) and (Field = nil) then
        begin
          StatusBar.Panels[3].Text :=

procedure TADODBTest.UseShapeProviderExecute(Sender: TObject);
begin
  BooleanActionExecute(Sender);
  Connection.Close;
end;

procedure TADODBTest.MaxRecordsExecute(Sender: TObject);
var
  MaxRecs: string;
begin
  MaxRecs := IntToStr(MaxRecords.Tag);
  if InputQuery(Application.Title, MaxRecords.Hint, MaxRecs) then
    MaxRecords.Tag := StrToInt(MaxRecs);
end;

{ Status Information }

procedure TADODBTest.ShowHeapStatus(Sender: TObject; var Done: Boolean);
begin
  Caption := Format('ADO DB Controls Test Application - (Blocks=%d Bytes=%d)',
    [AllocMemCount, AllocMemSize]);
end;

procedure TADODBTest.SetStatusMsg(const Msg: string);
begin
  StatusBar.Panels[0].Text := Msg;
end;

procedure TADODBTest.ShowProgressBar(const Msg: string);
begin
  ProgressBar.Show;
```

```
Field.Value) then
      StatusMsg := Format('Orignal Value: %s', [VarToStr(V)]) else
      StatusMsg := '';
    end;
  end;

begin
  Field := (Sender as TDBGrid).SelectedField;
  if Assigned(Field) then
  begin
    (Sender as TDBGrid).Hint := Field.ClassName;
    StatusBar.Panels[0].Text := Field.ClassName;
    StatusBar.Panels[2].Text := NullStr[Field.IsNull];
    TrackBlobs;
    if ActiveDataSet.UpdateStatus = usModified then
      ShowOriginalValues;
  end;
end;

{ Connection Operations }

procedure TADODBTest.CheckConnection(CloseFirst: Boolean);
const
  ConnectOptionValues: array [Boolean] of TConnectOption = (coConnectUnspecified, coAsyncConnect);
var
  ConnStr: string;
  Index: Integer;
begin
  if not CloseFirst and Connection.Connected then Exit;
  Connection.Close;
  MasterClientData.Close;
  ConnStr := ConnectionString.Text;
  Connection.ConnectionString := ConnStr;
  Connection.ConnectOptions := ConnectOptionValues[AsyncConnec
```

```
Format('%d of %d', [RecNo, RecordCount]);
        StatusMsg := '';
      end;
    end;
    if ActiveControl is TDBGrid then

GridColEnter(TDBGrid(ActiveControl));
    end;
  end;
  LogEvent('OnDataChange', Sender as TComponent);
end;

procedure TADODBTest.GridColEnter(Sender : TObject);
const
  NullStr: array[Boolean] of string = ('','[NULL]');
var
  Field: TField;

  procedure TrackBlobs;
  begin
    if Field.DataSet <> MasterDataSource.DataSet then Exit;
    if (Field is TMemoField) and (Field <> DBMemo1.Field) then
      DBMemo1.DataField := Field.FieldName
    else if (Field is TGraphicField) and (Field <> DBImage1.Field) then
      DBImage1.DataField := Field.FieldName;
  end;

  procedure ShowOriginalValues;
  var
    V: Variant;
  begin
    if Field.Dataset.CanModify then
    begin
      V := Field.OldValue;
      if not VarIsNull(V) and (V <>
```

```
procedure TADODBTest.ConnectionStringClick(Sender: TObject);
begin
  if (ConnectionString.Text <> '') and not ConnectionString.DroppedDown then
  begin
    CheckConnection(True);
    MasterTableName.Items.Clear;
    MasterTableName.Text := '';
    DetailTableName.Items.Clear;
    DetailTableName.Text := '';
    MasterProcName.Items.Clear;
    MasterProcName.Text := '';
    DetailProcName.Items.Clear;
    DetailProcName.Text := '';
  end;
end;

procedure TADODBTest.ConnectionStringKeyPress(Sender: TObject; var Key: Char);
begin
  if Key = #13 then
  begin
    if ConnectionString.DroppedDown then
      ConnectionString.DroppedDown := False;
    ConnectionStringClick(Sender);
    Key := #0;
  end;
end;

procedure TADODBTest.CloseConnectionExecute(Sender: TObject);
begin
  Connection.Close;
end;

{ Common DataSet Operations }
```

```
t.Checked];
  if UseShapeProvider.Checked then
    Connection.Provider := 'MSDataShape';
  Index := ConnectionString.Items.IndexOf(ConnStr);
  if Index > 0 then
    ConnectionString.Items.Delete(Index);
  if Index <> 0 then
  begin
    ConnectionString.Items.Insert(0, ConnStr);
    while ConnectionString.Items.Count > 20 do
      ConnectionString.Items.Delete(20);
  end;
  ConnectionString.ItemIndex := 0;
  Application.ProcessMessages;
  Connection.Open;
  if AsyncConnect.Checked and (stConnecting in Connection.State) then
    ShowProgressBar('Connecting');
// ShowProperties(Connection.Properties);
end;

procedure TADODBTest.EditConnStrClick(Sender: TObject);
begin
  Connection.Close;
  Connection.ConnectionString := ConnectionString.Text;
  if EditConnectionString(Connection) then
  begin
    ConnectionString.Text := Connection.ConnectionString;
    ConnectionStringClick(Sender);
  end;
end;
```

```
  ADOSource.Connection := nil;
end;

function TADODBTest.GetActiveDataSet: TDataSet;
begin
  if not Assigned(FActiveDataSet) then
    DatabaseError('No active dataset');
  Result := FActiveDataSet;
end;

procedure TADODBTest.SetActiveDataSet(Value: TDataSet);

  function GetDetailDataSet: TDataSet;
  var
    I: Integer;
  begin
    Result := nil;
    if (Value = MasterTable) and DetailTable.Active then
       Result := DetailTable
    else if (Value = MasterQuery) and DetailQuery.Active then
      Result := DetailQuery
    else if (Value = MasterProc) and DetailProc.Active then
      Result := DetailProc
    else for I := 0 to Value.Fields.Count - 1 do
      if Value.Fields[I] is TDataSetField then
      begin
        Result := TDataSetField(Value.Fields[I]).NestedDataSet;
        Break;
      end;
  end;

begin
  StatusBar.Panels[2].Text := '';
  MasterDataSource.Enabled :=
```

```
procedure TADODBTest.OpenDataSet(Source: TCustomADODataSet);

  procedure ShowFetchProgress;
  begin
    while stFetching in ADOSource.RecordSetState do
    begin
      with ADOSource do
        StatusBar.Panels[3].Text := Format('%d of %d', [RecNo, RecordCount]);
      Application.ProcessMessages;
    end;
  end;

begin
  ClearEventLog.Execute;
  Screen.Cursor := crHourGlass;
  try
    Source.Close;
    MasterClientData.Close;
    ADOSource := TCustomADODataSet(Source);
    SetRecordSetEvents(UseClientCursor.Checked, ADOSource);
    Provider.DataSet := ADOSource;
    if MidasButton.Checked then ActiveDataSet := MasterClientData else
    begin
      ActiveDataSet := Source;
      ShowFetchProgress;
    end;
    if MasterGrid.Visible then MasterGrid.SetFocus;
  finally
    Screen.Cursor := crDefault;
  end;
  StreamSettings(True);
end;

procedure TADODBTest.DisconnectDataSetExecute(Sender: TObject);
begin
```

```
  if Connection = nil then
    Connection := Self.Connection;
  CheckConnection(False);
  if UseClientCursor.Checked then
    CursorLocation := clUseClient
else
    CursorLocation := clUseServer;
  for I := 0 to
CursorTypeItem.Count - 1 do
  if
CursorTypeItem.Items[I].Checked
then
  begin
    CursorType := TCursortype(I);
    Break;
  end;
  for I := 0 to LockTypeItem.Count
- 1 do
  if
LockTypeItem.Items[I].Checked
then
  begin
    LockType := TADOLocktype(I);
    Break;
  end;
  ExecuteOptions := [];
  if AsyncExecute.Checked then
    ExecuteOptions :=
[eoAsyncExecute];
  if AsyncFetch.Checked then
    ExecuteOptions :=
ExecuteOptions +
[eoAsyncFetchNonBlocking];
  MaxRecords :=
Self.MaxRecords.Tag;
  EnableBCD :=
Self.EnableBCD.Checked;
 end;
 LogEvent('BeforeOpen', DataSet);
end;

procedure
TADODBTest.DataSetAfterOpen(D
ataSet: TDataSet);
begin
 ClearProgressBar;
//
ShowProperties(ADOSource.Record
```

```
False;
 DetailDataSource.Enabled := False;
 try
  MasterGrid.DataSource := nil;
  FActiveDataSet := Value;
  DetailDataSource.DataSet := nil;
  MasterDataSource.DataSet :=
Value;
  if Assigned(Value) then
  begin
   Value.Open;
   if AsyncExecute.Checked and
(Value.State = dsOpening) then
    ShowProgressBar('Executing');
   if DisplayDetails.Checked then
    DetailDataSource.DataSet :=
GetDetailDataSet;
  end;
  BindControls(Value);
 finally
  MasterDataSource.Enabled :=
True;
  DetailDataSource.Enabled :=
True;
 end;
 if Assigned(Value) then
 begin
  Update;
  StatusMsg := 'ActiveDataSet is ' +
Value.Name;
  if
Assigned(AreaSelector.ActivePage.O
nShow) then

AreaSelector.ActivePage.OnShow(ni
l);
 end;
end;

procedure
TADODBTest.DataSetBeforeOpen(
DataSet: TDataSet);
var
 I: Integer;
begin
 with DataSet as
TCustomADODataSet do
 begin
```

```
  end;
end;

procedure TADODBTest.MasterTableNameKeyPress(Sender: TObject; var Key: Char);
begin
  if Key = #13 then
  begin
    with Sender as TComboBox do
    if DroppedDown then DroppedDown := False;
    OpenTable.Execute;
    Key := #0;
  end;
end;

procedure TADODBTest.MasterTableNameClick(Sender: TObject);
begin
  with Sender as TComboBox do
  if not DroppedDown then
  begin
    DetailTableName.Text := '';
    OpenTable.Execute;
  end;
end;

procedure TADODBTest.DetailTableNameClick(Sender: TObject);
begin
  with Sender as TComboBox do
    if not DroppedDown and (DetailTable.TableName <> Text) then
      OpenTable.Execute;
end;

procedure TADODBTest.UpdateReOpenMenu;
var
  I: Integer;
begin
  while FileReOpen.Count > 0 do
    FileReOpen.Items[0].Free;
Set.Fields[0].Properties);
  LogEvent('AfterOpen', DataSet);
end;

procedure TADODBTest.DataSetAfterClose(DataSet: TDataSet);
begin
  LogEvent('AfterClose', DataSet);
  if DataSet = FActiveDataSet then
    FActiveDataSet := nil;
  if DataSet = ADOSource then
    FilterGroupBox.ItemIndex := -1;
end;

procedure TADODBTest.DataSetFetchComplete(DataSet: TCustomADODataSet;
  const Error: Error; var EventStatus: TEventStatus);
begin
  LogEvent('FetchComplete', DataSet);
end;

procedure TADODBTest.FormCloseQuery(Sender: TObject;
  var CanClose: Boolean);
begin
  CloseActiveDataSet.Execute;
end;

{ Table Operations }

procedure TADODBTest.MasterTableNameDropDown(Sender: TObject);
begin
  try
    CheckConnection(False);
    with Sender as TComboBox do
      if Items.Count < 1 then
        Connection.GetTableNames(Items);
  except
    { Eat any exceptions so the combobox doesn't paint funny }
```

```
procedure TADODBTest.CloseActiveDataSetExecute(Sender: TObject);
begin
  ActiveDataSet.Close;
end;

procedure TADODBTest.MasterTableBeforeOpen(DataSet: TDataSet);
begin
  DataSetBeforeOpen(DataSet);
  MasterTable.TableDirect := UseTableDirect.Checked;
  MasterTable.TableName := MasterTableName.Text;
  DetailTable.MasterSource := nil;
end;

procedure TADODBTest.DetailTableBeforeOpen(DataSet: TDataSet);
begin
  DataSetBeforeOpen(DataSet);
  DetailTable.TableDirect := UseTableDirect.Checked;
  DetailTable.TableName := DetailTableName.Text;
end;

procedure TADODBTest.MasterTableAfterOpen(DataSet: TDataSet);
var
  I: Integer;
  Field: TField;
  MasterFields: string;
begin
  if DetailTableName.Text <> '' then
  begin
    DetailTable.Open;
    if MasterTableName.Text = DetailTableName.Text then
      MasterFields := MasterTable.Fields[0].FieldName+'; '
    else
```

```
  for I := 0 to FClosedTables.Count - 1 do
    FileReOpen.Add(NewItem(Format('%d) %s', [I, FClosedTables[I]]), 0, False,
      True, ClosedFileClick, 0, ''));
end;

procedure TADODBTest.ClosedFileClick(Sender: TObject);
var
  S: string;
  Index, P, P2: Integer;
begin
  S := Copy(TMenuItem(Sender).Caption, 5, MAXINT);
  P := Pos(':', S);
  P2 := Pos('/', S);
  if P2 > 0 then
    DetailTableName.Text := Copy(S, P2+1, MAXINT) else
  begin
    DetailTableName.Text := '';
    P2 := MAXINT;
  end;
  MasterTableName.Text := Copy(S, P+1, P2-P-1);
  Index := FClosedTables.IndexOf(S);
  if Index > -1 then
    FClosedTables.Delete(Index);
  OpenTable.Execute;
end;

procedure TADODBTest.OpenTableExecute(Sender: TObject);
begin
  if MasterTableName.Text <> '' then
  begin
    MasterTable.Close;
    OpenDataSet(MasterTable);
  end;
end;
```

```
MasterTable.TableName;
  if DetailTable.Active then
  begin
    TableEntry := TableEntry + '/' + DetailTable.TableName;
    DetailTable.Close;
  end;
  if FClosedTables.IndexOf(TableEntry) = -1 then
  begin
    FClosedTables.Insert(0, TableEntry);
    if FClosedTables.Count > 9 then
      FClosedTables.Delete(9);
  end;
 end;

begin
 UpdateClosedTables;
 DetailTable.Close;
 DataSetBeforeClose(Dataset);
end;

{ Query Operations }

procedure TADODBTest.ProcessQuery(SelectQuery: Boolean);

 procedure UpdateQueryHistory;
 var
  DSQL: string;
 begin
  if FMasterQueries.IndexOf(MasterSQL.Text) <> -1 then Exit;
  FMasterQueries.Add(MasterSQL.Text);
  DSQL := DetailSQL.Text;
  if DSQL = '' then DSQL := '(empty)';
  FDetailQueries.Insert(0, DSQL);
  if FMasterQueries.Count > 9 then
  begin
    FMasterQueries.Delete(0);
    FDetailQueries.Delete(0);
```

```
    for I := 0 to DetailTable.Fields.Count - 1 do
    begin
      Field := MasterTable.FindField(DetailTable.Fields[I].FieldName);
      if Field <> nil then
      begin
        if DetailTable.IndexDefs.GetIndexForFields(MasterFields + Field.FieldName, False) <> nil then
          MasterFields := MasterFields + Field.FieldName + ';';
      end;
    end;
   if MasterFields = '' then
     DatabaseError('Cannot determine linking fields for detail');
   SetLength(MasterFields, Length(MasterFields)-1);
   DetailTable.IndexFieldNames := MasterFields;
   DetailTable.MasterFields := MasterFields;
   DetailTable.MasterSource := DetailMasterSource;
  end;
  DataSetAfterOpen(DataSet);
end;

procedure TADODBTest.FileMenuClick(Sender: TObject);
begin
  UpdateReOpenMenu;
  FileReOpen.Enabled := FClosedTables.Count > 0;
end;

procedure TADODBTest.MasterTableBeforeClose(DataSet: TDataSet);

  procedure UpdateClosedTables;
  var
   TableEntry: string;
  begin
   TableEntry :=
```

```
end;

procedure TADODBTest.ExecSQLExecute(Sender: TObject);
begin
  ProcessQuery(False);
end;

procedure TADODBTest.OpenQueryExecute(Sender: TObject);
begin
  ProcessQuery(True);
end;

procedure TADODBTest.PrevQueryUpdate(Sender: TObject);
begin
  PrevQuery.Enabled := FQueryIndex < (FMasterQueries.Count - 1);
end;

procedure TADODBTest.PrevQueryExecute(Sender: TObject);
begin
  Assert(FQueryIndex < (FMasterQueries.Count - 1));
  Inc(FQueryIndex);
  SetQueryText;
end;

procedure TADODBTest.NextQueryExecute(Sender: TObject);
begin
  if FQueryIndex > -1 then
    Dec(FQueryIndex);
  SetQueryText;
end;

procedure TADODBTest.MasterSQLKeyPress(Sender: TObject; var Key: Char);
begin
```

```
    end;
  end;

var
  RecordsAffected: Integer;
begin
  CheckConnection(False);
  if SelectQuery then
  begin
    MasterQuery.Close;
    MasterQuery.SQL.Text := MasterSQL.Text;
    WriteParameterData;
    OpenDataSet(MasterQuery)
  end else
  begin
    if SQLParams.Checked then
    begin
      ADOCommand.CommandType := cmdText;
      ADOCommand.CommandText := MasterSQL.Text;
    end else
    begin
      ADOCommand.CommandType := cmdStoredProc;
      ADOCommand.CommandText := MasterProcName.Text;
    end;
    if ParameterList.Items.Count > 0 then
    begin
      WriteParameterData;
      ADOCommand.Parameters.Assign(FParamSource);
    end;
    ADOCommand.Execute(RecordsAffected, EmptyParam);
    StatusMsg := Format('%d rows were affected', [RecordsAffected]);
    if ProcParams.Checked then
      MasterProc.Parameters.Assign(ADOCommand.Parameters);
  end;
  UpdateQueryHistory;
```

```
MasterSQL.Text;
end;

procedure TADODBTest.DetailQueryBeforeOpen(DataSet: TDataSet);
begin
  DataSetBeforeOpen(DataSet);
  DetailQuery.SQL.Text := DetailSQL.Text;
  DetailQuerySource.Dataset := MasterQuery;
  if DetailQuery.Parameters.Count = 0 then
    RefreshParameters(DetailQuery.Parameters);
end;

procedure TADODBTest.MasterQueryAfterOpen(DataSet: TDataSet);
begin
  if Trim(DetailSQL.Text) <> '' then
    DetailQuery.Open else
    DetailQuerySource.Dataset := nil;
  DataSetAfterOpen(DataSet);
end;

procedure TADODBTest.MasterQueryBeforeClose(DataSet: TDataSet);
begin
  DetailQuery.Close;
  DataSetBeforeClose(DataSet);
end;

{ Stored Procedures }

procedure TADODBTest.MasterProcBeforeOpen(DataSet: TDataSet);
begin
  DataSetBeforeOpen(DataSet);
  MasterProc.ProcedureName := MasterProcName.Text;
  WriteParameterData;
end;
```

```
  if Key = #13 then
  begin
    OpenQuery.Execute;
    Key := #0;
  end;
end;

procedure TADODBTest.SetQueryText;
var
  DSQL: string;
begin
  if FQueryIndex > -1 then
  begin
    MasterSQL.Text := FMasterQueries[FQueryIndex];
    DSQL := FDetailQueries[FQueryIndex];
    if DSQL = '(empty)' then DSQL := '';
    DetailSQL.Text := DSQL;
  end else
  begin
    MasterSQL.Text := '';
    DetailSQL.Text := '';
  end;
end;

procedure TADODBTest.EditCommandTextClick(Sender: TObject);
var
  Command: string;
begin
  CheckConnection(False);
  Command := MasterSQL.Text;
  if EditSQL(Command, Connection.GetTableNames, Connection.GetFieldNames) then
    MasterSQL.Text := Command;
end;

procedure TADODBTest.MasterQueryBeforeOpen(DataSet: TDataSet);
begin
  DataSetBeforeOpen(DataSet);
  MasterQuery.SQL.Text :=
```

```
  var Key: Char);
begin
  if Key = #13 then
  begin
    with Sender as TComboBox do
    if DroppedDown then
DroppedDown := False;
    Key := #0;
  end;

end;

procedure
TADODBTest.MasterProcNameClic
k(Sender: TObject);
begin
  with Sender as TComboBox do
  if not DroppedDown then
    DetailProcName.Text := '';
end;

procedure
TADODBTest.DetailProcNameClick
(Sender: TObject);
begin
end;

{ Packet Save/Load }

procedure
TADODBTest.LoadFromFileExecut
e(Sender: TObject);
begin
  OpenDialog.FilterIndex := 1;
  OpenDialog.FileName :=
FLastDataFile;
  if OpenDialog.Execute then
  begin

ADODataSet.LoadFromFile(OpenDi
alog.FileName);
    FLastDataFile :=
OpenDialog.FileName;
    ActiveDataSet := ADODataSet;
    ADOSource := ADODataSet;
  end;
end;

```

```

procedure
TADODBTest.MasterProcAfterOpe
n(DataSet: TDataSet);
begin
  if DetailProcName.Text <> '' then
    DetailProc.Open;
  DataSetAfterOpen(DataSet);
end;

procedure
TADODBTest.DetailProcBeforeOpe
n(DataSet: TDataSet);
begin
  DataSetBeforeOpen(DataSet);
  DetailProc.ProcedureName :=
DetailProcName.Text;

RefreshParameters(DetailProc.Para
meters);
end;

procedure
TADODBTest.OpenProcedureExecu
te(Sender: TObject);
begin
  if MasterProcName.Text <> '' then
  begin
    MasterProc.Close;
    OpenDataSet(MasterProc);
  end;
end;

procedure
TADODBTest.ProcNameDropDown
(Sender: TObject);
begin
  CheckConnection(False);
  with Sender as TComboBox do
    if Items.Count < 1 then

Connection.GetProcedureNames(Ite
ms);
end;

procedure
TADODBTest.MasterProcNameKey
Press(Sender: TObject;
```

```
end;

procedure
TADODBTest.StreamFormInClick(
Sender: TObject);
var
  Form: TADODBTest;
begin
  OpenDialog.FilterIndex := 2;
  OpenDialog.FileName :=
FLastFormFile;
  if OpenDialog.Execute then
  begin
    Form :=
TADODBTest.CreateNew(Applicati
on, 0);

ReadComponentResFile(OpenDialo
g.FileName, Form);
    FLastFormFile :=
OpenDialog.FileName;
    Form.FormCreate(Form);
  end;
end;

{ DB Control Linking }

procedure
TADODBTest.BindControls(DataSe
t: TDataSet);

  procedure DeleteDBEditControls;
  var
    I: Integer;
  begin
    with DBEditScroller do
      for I := ComponentCount - 1
downto 0 do
        if (Components[I] is TDBEdit)
or (Components[I] is TLabel) then
          Components[I].Free;
  end;

  procedure
SetDisplayType(ForwardOnly:
Boolean);
  begin
    if ForwardOnly then
```

```
procedure
TADODBTest.SaveToFileExecute(S
ender: TObject);
begin
  SaveDialog.FilterIndex := 1;
  SaveDialog.FileName :=
FLastDataFile;
  if SaveDialog.Execute then
  begin

ADOSource.SaveToFile(SaveDialog.
FileName, pfADTG);
    FLastDataFile :=
SaveDialog.FileName;
  end;
end;

procedure
TADODBTest.FileActionsUpdate(Se
nder: TObject);
begin
  SaveToFile.Enabled :=
Assigned(FActiveDataSet) and
ActiveDataSet.Active;
  CloseActiveDataSet.Enabled :=
SaveToFile.Enabled;
  DisconnectDataset.Enabled :=
SaveToFile.Enabled;
  CloseConnection.Enabled :=
Connection.Connected;
end;

{ Streaming }

procedure
TADODBTest.StreamFormOutClick
(Sender: TObject);
begin
  SaveDialog.FilterIndex := 2;
  SaveDialog.FileName :=
FLastFormFile;
  if SaveDialog.Execute then
  begin

WriteComponentResFile(SaveDialog
.FileName, Self);
    FLastFormFile :=
SaveDialog.FileName;
  end;
```

```
    begin
      AutoSize := False;
      Alignment := taRightJustify;
      Left := 3;
      Top := LabelTop;
      Width := 59;
      Parent := DBEditScroller;
      Caption := F.DisplayLabel+':';
    end;
  end;

var
  I: Integer;
  Field: TField;
begin
  DBMemo1.DataField := '';
  DBImage1.DataField := '';
  DetailGrid.Visible := False;
  GridSplitter.Visible := False;
  BlobCtrlPanel.Visible := False;
  DBImage1.Visible := False;
  DBMemo1.Visible := False;
  ReadOnlyLabel.Visible := False;
  if Assigned(DataSet) and
DataSet.Active then
  begin
    SetDisplayType((DataSet is
TCustomADODataSet) and not

TCustomADODataSet(DataSet).Sup
ports([coBookmark,
coMovePrevious]));
    for I := 0 to DataSet.FieldCount -
1 do
    begin
      Field := DataSet.Fields[I];
      Field.OnValidate :=
FieldValidate;
      case Field.DataType of
        ftMemo:
          if DBMemo1.DataField = ''
then
          begin
            DBMemo1.DataField :=
Field.FieldName;
            DBMemo1.Visible := True;
          end;
        ftGraphic:
```

```
    begin
      MasterGrid.Visible := False;
      MasterGrid.DataSource := nil;
      DBEditScroller.Height :=
GridPanel.Height;

DBEditScroller.HorzScrollBar.Positi
on := 0;

DBEditScroller.VertScrollBar.Positi
on := 0;
      DBNavigator1.VisibleButtons :=
[nbNext, nbInsert, nbDelete, nbEdit,
        nbPost, nbCancel, nbRefresh];
    end else
    begin
      MasterGrid.Visible := True;
      MasterGrid.DataSource :=
MasterDataSource;
      DBEditScroller.Height := 0;
      DBNavigator1.VisibleButtons :=
[nbFirst, nbPrior, nbNext, nbLast,
        nbInsert, nbDelete, nbEdit,
nbPost, nbCancel, nbRefresh];
    end;
  end;

  procedure CreateDBEdit(F:
TField);
  var
    LabelTop: Integer;
  begin
    with
TDBEdit.Create(DBEditScroller) do
    begin
      Left := 65;
      Top := (F.FieldNo - 1) * (Height
+ 5) + 5;
      LabelTop := Top + 3;
      Width := F.DisplayWidth *
Canvas.TextWidth('0');
      Parent := DBEditScroller;
      DataSource :=
MasterDataSource;
      DataField := F.FieldName;
    end;
    with
TLabel.Create(DBEditScroller) do
```

```
: TObject);
begin
  if not Assigned(FActiveDataSet) then Exit;
  ActiveDataSource := (Sender as TDBGrid).DataSource;
  DBNavigator1.DataSource := ActiveDataSource;
  DataSourceDataChange(ActiveDataSource, nil);
end;

procedure TADODBTest.PopupMenu1Popup(Sender: TObject);
var
  I: Integer;
  MI: TMenuItem;
  F, CurField: TField;
begin
  with PopupMenu1, ActiveDataSet do
  begin
    if PopupMenu1.PopupComponent = DBMemo1 then
      CurField := DBMemo1.Field else
      CurField := DBImage1.Field;
    while Items.Count > 0 do Items.Delete(0);
    MI := NewItem('(None)', 0, False, True, FieldSelect, 0, 'None');
    Items.Add(MI);
    for I := 0 to FieldCount - 1 do
      if Fields[I] is TBlobField then
      begin
        F := Fields[I];
        MI := NewItem(F.FieldName, 0, F=CurField, True, FieldSelect, 0, 'mi'+F.FieldName);
        MI.Tag := Integer(F);
        Items.Add(MI);
      end;
  end;
end;

procedure TADODBTest.FieldSelect(Sender:
```

```
        if DBImage1.DataField = '' then
        begin
          DBImage1.DataField := DataSet.Fields[I].FieldName;
          DBImage1.Visible := True;
        end;
      ftDataSet, ftReference:
        if DisplayDetails.Checked and (DetailDataSource.DataSet = nil) then
        begin
          DetailDataSource.DataSet := TDataSetField(DataSet.Fields[I]).NestedDataSet;
        end;
      ftBytes, ftVarBytes:
        begin
          Field.OnGetText := BinaryGetText;
          Field.OnSetText := BinarySetText;
          Field.DisplayWidth := (Field.Size + 3);
        end;
      else
        if not MasterGrid.Visible then
          CreateDBEdit(Field);
    end;
  end;
  BlobCtrlPanel.Visible := DBMemo1.Visible or DBImage1.Visible;
  ReadOnlyLabel.Visible := not DataSet.CanModify;
  if Assigned(DetailDataSource.DataSet) then
  begin
    GridSplitter.Visible := True;
    DetailGrid.Visible := True;
  end;
 end else
  DeleteDBEditControls;
end;

procedure TADODBTest.GridSetFocus(Sender
```

```
  MidasApplyUpdates.Enabled := MasterClientData.Active and
    ((MasterClientData.ChangeCount > 0) or (MasterClientData.State in dsEditModes));
  MidasCancelUpdates.Enabled := MidasApplyUpdates.Enabled;
  LoadBlobFromFile.Enabled := Enabled and (MasterGrid.SelectedField is TBlobField);
end;

procedure TADODBTest.BatchUpdateExecute(Sender: TObject);
begin
  if ADOSource.Connection = nil then
  begin
    CheckConnection(False);
    ADOSource.Connection := Connection;
  end;
  ADOSource.UpdateBatch;
end;

procedure TADODBTest.CancelBatchExecute(Sender: TObject);
begin
  ADOSource.CancelUpdates;
end;

procedure TADODBTest.ClearFieldExecute(Sender: TObject);
var
  Field: TField;
begin
  Field := MasterGrid.SelectedField;
  if Field = nil then Exit;
  ActiveDataSet.Edit;
  Field.Clear;
end;

procedure TADODBTest.RefreshDataExecute(Sender: TObject);
var
  MI: TMenuItem;
begin
  MI := TMenuItem(Sender);
  if PopupMenu1.PopupComponent = DBImage1 then
  try
    if MI.Tag = 0 then
      DBImage1.DataField := '' else
      DBImage1.DataField := TField(MI.Tag).FieldName;
  except
    DBImage1.DataField := '';
    raise;
  end
  else if PopupMenu1.PopupComponent = DBMemo1 then
  try
    if MI.Tag = 0 then
      DBMemo1.DataField := '' else
      DBMemo1.DataField := TField(MI.Tag).FieldName;
  except
    DBMemo1.DataField := '';
    raise;
  end;
end;

{ Editing / Updates }

procedure TADODBTest.EditActionsUpdate(Sender: TObject);
var
  Enabled: Boolean;
begin
  Enabled := Assigned(FActiveDataSet);
  BatchUpdate.Enabled := Assigned(ADOSource) and
    (ADOSource.LockType = ltBatchOptimistic);
  CancelBatch.Enabled := BatchUpdate.Enabled;
  ClearField.Enabled := Enabled;
  RefreshData.Enabled := Enabled;
```

```
TBlobField(MasterGrid.SelectedFiel
d).LoadFromFile(OpenDialog.FileN
ame);
end;

{ Indexes }

procedure
TADODBTest.IndexPageShow(Send
er: TObject);
begin
  if not (Assigned(ActiveDataSource)
and
Assigned(ActiveDataSource.DataSet
)) then Exit;
  RefreshIndexNames;
end;

procedure
TADODBTest.RefreshIndexNames;
var
  I: Integer;
  IndexDefs: TIndexDefs;
begin
  IndexList.Clear;
  if ActiveDataSet =
MasterClientData then
    IndexDefs :=
MasterClientData.IndexDefs else
  if ADOSource is TADOTable then
    IndexDefs :=
TADOTable(ADOSource).IndexDefs
else
  if ADOSource is TADODataSet
then
    IndexDefs :=
TADODataSet(ADOSource).IndexD
efs
  else
    Exit;
  IndexDefs.Update;
  for I := 0 to IndexDefs.Count - 1 do
    if IndexDefs[I].Name = '' then

IndexList.Items.Add('<primary>')
else

IndexList.Items.Add(IndexDefs[I].N
```

```
begin
  ActiveDataSet.Refresh;
end;

procedure
TADODBTest.BinaryGetText(Sende
r: TField; var Text: string;
  DisplayText: Boolean);
begin
  Text := Sender.AsString;
end;

procedure
TADODBTest.BinarySetText(Sende
r: TField; const Text: string);
begin
  Sender.AsString := Text;
end;

procedure
TADODBTest.BlobAsImageUpdate(
Sender: TObject);
begin
  BlobAsImage.Enabled :=
Assigned(ActiveDataSource.DataSet
) and
ActiveDataSource.DataSet.Active
and
    (MasterGrid.SelectedField is
TBlobField);
end;

procedure
TADODBTest.BlobAsImageExecute
(Sender: TObject);
begin
  BlobCtrlPanel.Visible := True;
  DBImage1.Visible := True;
  DBImage1.DataField :=
MasterGrid.SelectedField.FieldNam
e;
end;

procedure
TADODBTest.LoadBlobFromFileEx
ecute(Sender: TObject);
begin
  OpenDialog.FilterIndex := 3;
  if OpenDialog.Execute then
```

```
end;

procedure TADODBTest.IndexListClick(Sender: TObject);
begin
  ShowIndexParams;
  if ActiveDataSet is TADOTable then
    with TADOTable(ActiveDataSet) do
    begin
      try
        { Only Jet 4 supports setting indexname while open }
        MasterTable.IndexName := IndexList.Items[IndexList.ItemIndex];
      except
        Close;
        MasterTable.IndexName := IndexList.Items[IndexList.ItemIndex];
        OpenTableExecute(nil);
      end;
    end;
end;

procedure TADODBTest.GridTitleClick(Column: TColumn);
var
  DataSet: TDataSet;
begin
  if not FMovingColumn then
  begin
    DataSet := Column.Field.DataSet;
    if DataSet is TCustomADODataSet then
    with TCustomADODataSet(DataSet) do
    begin
      if (Pos(Column.Field.FieldName, Sort) = 1) and (Pos(' DESC', Sort) = 0) then
        Sort := Column.Field.FieldName + ' DESC' else
        Sort :=
```

```
ame);
  if IndexList.Items.Count > 0 then
  begin
    if (ADOSource = MasterTable) and (IndexList.Items.IndexOf(MasterTable.IndexName) > 0) then
      IndexList.ItemIndex := IndexList.Items.IndexOf(MasterTable.IndexName) else
      IndexList.ItemIndex := 0;
    ShowIndexParams;
  end;
end;

procedure TADODBTest.ShowIndexParams;
var
  IndexDef: TIndexDef;
begin
  if ActiveDataSource.DataSet is TADOTable then
    IndexDef := TADOTable(ActiveDataSource.DataSet).IndexDefs[IndexList.ItemIndex] else
  if ActiveDataSource.DataSet is TADODataSet then
    IndexDef := TADODataSet(ActiveDataSource.DataSet).IndexDefs[IndexList.ItemIndex]
  else
    Exit;
  idxCaseInsensitive.Checked := ixCaseInsensitive in IndexDef.Options;
  idxDescending.Checked := ixDescending in IndexDef.Options;
  idxUnique.Checked := ixUnique in IndexDef.Options;
  idxPrimary.Checked := ixPrimary in IndexDef.Options;
  IndexFields.Text := IndexDef.Fields;
  DescFields.Text := IndexDef.DescFields;
  CaseInsFields.Text := IndexDef.CaseInsFields;
```

```
Format('%s=%s%s%1:s',
[Field.FullName, QuoteChar,
LocValue]);
  end;
end;

procedure
TADODBTest.FilterKeyPress(Sende
r: TObject; var Key: Char);
begin
  FilterGroupBox.ItemIndex := -1;
  if Key = #13 then
FilteredClick(Sender);
end;

procedure
TADODBTest.FilterExit(Sender:
TObject);
begin
  if Assigned(FActiveDataSet) then
    ActiveDataSet.Filter :=
Filter.Text;
end;

procedure
TADODBTest.FilteredClick(Sender:
TObject);
begin
  if Filtered.Checked then
    ActiveDataSet.Filter :=
Filter.Text;
  ActiveDataSet.Filtered :=
Filtered.Checked;
end;

procedure
TADODBTest.FindFirstClick(Sende
r: TObject);
begin
  ActiveDataSet.Filter := Filter.Text;
  ActiveDataSet.FindFirst;
end;

procedure
TADODBTest.FindNextClick(Sende
r: TObject);
begin
  if ActiveDataSet.Filter <>
Filter.Text then
```

```
Column.Field.FieldName + ' ASC';
      StatusMsg := 'Sorted on '+Sort;
    end;
  end;
  FMovingColumn := False;
end;

procedure
TADODBTest.MasterGridColumn
Moved(Sender: TObject;
FromIndex,
  ToIndex: Integer);
begin
  FMovingColumn := True;
end;

{ Filters }

procedure
TADODBTest.FilterPageShow(Send
er: TObject);
var
  Field: TField;
  LocValue,
  QuoteChar: string;
begin
  if (Filter.Text = '') and
Assigned(FActiveDataSet) and
ActiveDataSet.Active then
  begin
    Field :=
MasterGrid.SelectedField;
    if Field = nil then Exit;
    with ActiveDataSet do
    try
      DisableControls;
      MoveBy(3);
      LocValue :=
VarToStr(Field.Value);
      First;
    finally
      EnableControls;
    end;
    if Field.DataType in [ftString,
ftMemo, ftFixedChar] then
      QuoteChar := '''' else
      QuoteChar := '';
    Filter.Text :=
```

```
Field.FieldName;
  with ActiveDataSet do
  try
    DisableControls;
    MoveBy(3);
    LocateEdit.Text :=
VarToStr(Field.Value);
    First;
  finally
    EnableControls;
  end;
 end;
end;

procedure
TADODBTest.LocateFieldDropDow
n(Sender: TObject);
begin

ActiveDataSet.GetFieldNames(Locat
eField.Items);
end;

procedure
TADODBTest.LocateButtonClick(Se
nder: TObject);

 function LocateValue: Variant;
 var
  I: Integer;
  Values: TStringList;
 begin
  if LocateNull.Checked then
    Result := Null
  else if Pos(',', LocateEdit.Text) < 1
then
    LocateValue := LocateEdit.Text
  else
  begin
    Values := TStringList.Create;
    try
      Values.CommaText :=
LocateEdit.Text;
      Result :=
VarArrayCreate([0,Values.Count-1],
varVariant);
      for I := 0 to Values.Count - 1 do
        Result[I] := Values[I];
```

```
  ActiveDataSet.Filter :=
Filter.Text;
 ActiveDataSet.FindNext;
end;

procedure
TADODBTest.FilterGroupBoxClick
(Sender: TObject);
begin
 if not Assigned(ADOSource) then
Exit;
 case FilterGroupBox.ItemIndex of
  0: ADOSource.FilterGroup :=
fgPendingRecords;
  1: ADOSource.FilterGroup :=
fgAffectedRecords;
  2: ADOSource.FilterGroup :=
fgFetchedRecords;
  3: ADOSource.FilterGroup :=
fgPendingRecords;
 else
  ADOSource.FilterGroup :=
fgNone;
 end;
end;


{ Locate }

procedure
TADODBTest.LocatePageShow(Sen
der: TObject);
var
 Field: TField;
begin
 if (FActiveDataSet <> nil) and
ActiveDataSet.Active then
 begin
  Field :=
MasterGrid.SelectedField;
  if LocateField.Items.Count = 0
then

LocateFieldDropDown(LocateField)
;
  if (LocateField.Text = '') or
(LocateField.Items.IndexOf(Field.Fi
eldName) < 1) then
    LocateField.Text :=
```

xecute(Sender: TObject);
begin
StatusMsg := 'ApplyUpdates: '+ IntToStr(MasterClientData.ApplyUpdates(-1));
Beep;
end;
procedure TADODBTest.MidasCancelUpdatesExecute(Sender: TObject);
begin
MasterClientData.CancelUpdates;
StatusMsg := 'Updates canceled';
end;
procedure TADODBTest.MasterClientDataReconcileError(
DataSet: TCustomClientDataSet; E: EReconcileError; UpdateKind: TUpdateKind;
var Action: TReconcileAction);
begin
Action := HandleReconcileError(DataSet, UpdateKind, E);
end;
{ FieldSchema }
procedure TADODBTest.FieldsPageShow(Sender: TObject);
begin
CheckConnection(False);
Connection.OpenSchema(siColumns, VarArrayOf([Unassigned, Unassigned, MasterTableName.Text, Unassigned]), EmptyParam, FieldSchema);
end;
procedure TADODBTest.FieldSchemaDATA_TYPEGetText(Sender: TField;
var Text: String; DisplayText:

finally
Values.Free;
end;
end;
end;
var
Options: TLocateOptions;
begin
Options := [];
if locPartialKey.Checked then Include(Options, loPartialKey);
if ActiveDataSet.Locate(LocateField.Text, LocateValue, Options) then
StatusMsg := 'Record Found' else
StatusMsg := 'Not found';
end;
procedure TADODBTest.LocateNullClick(Sender: TObject);
begin
LocateEdit.Enabled := not LocateNull.Checked;
end;
{ Midas Testing }
procedure TADODBTest.ADOButtonClick(Sender: TObject);
begin
ActiveDataSet := ADOSource;
end;
procedure TADODBTest.MidasButtonClick(Sender: TObject);
begin
if Assigned(ADOSource) or MasterClientData.Active then
ActiveDataSet := MasterClientData;
end;
procedure TADODBTest.MidasApplyUpdatesE

```
'adVarBinary';
   $000000CD: Text :=
'adLongVarBinary';
   $00000088: Text := 'adChapter';
   $00000040: Text := 'adFileTime';
   $00000089: Text :=
'adDBFileTime';
   $0000008A: Text :=
'adPropVariant';
   $0000008B: Text :=
'adVarNumeric';
  else
   Text := '<Unknown>';
  end;

end;

{ Event Logging }

procedure
TADODBTest.LogEvent(const
EventStr: string;
  Component: TComponent = nil);
var
  ItemCount: Integer;
begin
  if (csDestroying in
ComponentState) or not
Events.Visible then Exit;
  if (Component <> nil) and
(Component.Name <> '') then

Events.Items.Add(Format('%s(%s)',
[EventStr, Component.Name])) else
   Events.Items.Add(EventStr);
  ItemCount := Events.Items.Count;
  Events.ItemIndex := ItemCount - 1;
  if ItemCount >
(Events.ClientHeight div
Events.ItemHeight) then
   Events.TopIndex := ItemCount -
1;
end;

procedure
TADODBTest.ClearEventLogExecu
te(Sender: TObject);
begin
```

```
Boolean);
begin
  case
FieldSchemaData_TYPE.Value of
   $00000000: Text := 'adEmpty';
   $00000010: Text := 'adTinyInt';
   $00000002: Text := 'adSmallInt';
   $00000003: Text := 'adInteger';
   $00000014: Text := 'adBigInt';
   $00000011: Text :=
'adUnsignedTinyInt';
   $00000012: Text :=
'adUnsignedSmallInt';
   $00000013: Text :=
'adUnsignedInt';
   $00000015: Text :=
'adUnsignedBigInt';
   $00000004: Text := 'adSingle';
   $00000005: Text := 'adDouble';
   $00000006: Text := 'adCurrency';
   $0000000E: Text := 'adDecimal';
   $00000083: Text := 'adNumeric';
   $0000000B: Text := 'adBoolean';
   $0000000A: Text := 'adError';
   $00000084: Text :=
'adUserDefined';
   $0000000C: Text := 'adVariant';
   $00000009: Text := 'adIDispatch';
   $0000000D: Text :=
'adIUnknown';
   $00000048: Text := 'adGUID';
   $00000007: Text := 'adDate';
   $00000085: Text := 'adDBDate';
   $00000086: Text := 'adDBTime';
   $00000087: Text :=
'adDBTimeStamp';
   $00000008: Text := 'adBSTR';
   $00000081: Text := 'adChar';
   $000000C8: Text := 'adVarChar';
   $000000C9: Text :=
'adLongVarChar';
   $00000082: Text := 'adWChar';
   $000000CA: Text :=
'adVarWChar';
   $000000CB: Text :=
'adLongVarWChar';
   $00000080: Text := 'adBinary';
   $000000CC: Text :=
```

```
begin
  LogEvent('BeforeCancel');
end;

procedure
TADODBTest.DataSetBeforeDelete(
DataSet: TDataSet);
begin
  LogEvent('BeforeDelete', DataSet);
end;

procedure
TADODBTest.DataSetBeforeEdit(D
ataSet: TDataSet);
begin
  LogEvent('BeforeEdit', DataSet);
end;

procedure
TADODBTest.DataSetBeforeInsert(
DataSet: TDataSet);
begin
  LogEvent('BeforeInsert', DataSet);
end;

procedure
TADODBTest.DataSetBeforePost(D
ataSet: TDataSet);
begin
  LogEvent('BeforePost', DataSet);
end;

procedure
TADODBTest.DataSetBeforeScroll(
DataSet: TDataSet);
begin
  LogEvent('BeforeScroll', DataSet);
end;

procedure
TADODBTest.DataSetCalcFields(Da
taSet: TDataSet);
begin
  LogEvent('OnCalcFields',
DataSet);
end;

procedure
```

```
  Events.Items.Clear;
end;

procedure
TADODBTest.ClearEventLogUpdat
e(Sender: TObject);
begin
  ClearEventLog.Enabled :=
Events.Visible and
(Events.Items.Count > 0);
end;

procedure
TADODBTest.SetRecordSetEvents(
Hook: Boolean; DataSet:
TCustomADODataSet);
begin
  if Hook then
  begin
    DataSet.OnFetchComplete :=
DataSetFetchComplete;
    DataSet.OnFetchProgress :=
DataSetFetchProgress;
  end
  else
  begin
    DataSet.OnFetchComplete := nil;
    DataSet.OnFetchProgress := nil;
  end;
end;

procedure
TADODBTest.DataSetBeforeClose(
DataSet: TDataSet);
begin
  LogEvent('BeforeClose');
end;

procedure
TADODBTest.DataSetAfterScroll(D
ataSet: TDataSet);
begin
  LogEvent('AfterScroll', DataSet);
end;

procedure
TADODBTest.DataSetBeforeCancel
(DataSet: TDataSet);
```

```
begin
  LogEvent('AfterCancel', DataSet);
end;

procedure
TADODBTest.DataSourceStateChange(Sender: TObject);
begin
  LogEvent('OnStateChange',
Sender as TComponent);
end;

procedure
TADODBTest.DataSourceUpdateData(Sender: TObject);
begin
  LogEvent('OnUpdateData', Sender
as TComponent);
end;

procedure
TADODBTest.MasterTableBeforeScroll(DataSet: TDataSet);
begin
  LogEvent('BeforeScroll', DataSet);
end;

procedure
TADODBTest.MasterTableAfterScroll(DataSet: TDataSet);
begin
  LogEvent('AfterScroll', DataSet);
end;

procedure
TADODBTest.OnFilterRecord(DataSet: TDataSet; var Accept:
Boolean);
begin
  Accept :=
(DataSet.Fields[0].AsInteger = 2);
end;

procedure
TADODBTest.DataSetFetchProgress(DataSet: TCustomADODataSet;
  Progress, MaxProgress: Integer;
var EventStatus: TEventStatus);
begin
```

```
TADODBTest.DataSetError(DataSet: TDataSet;
  E: EDatabaseError; var Action:
TDataAction);
begin

LogEvent('OnDelete/OnEdit/OnPost
Errors', DataSet);
end;

procedure
TADODBTest.DataSetNewRecord(DataSet: TDataSet);
begin
  LogEvent('OnNewRecord',
DataSet);
end;

procedure
TADODBTest.DataSetAfterPost(DataSet: TDataSet);
begin
  LogEvent('AfterPost', DataSet);
end;

procedure
TADODBTest.DataSetAfterInsert(DataSet: TDataSet);
begin
  LogEvent('AfterInsert', DataSet);
end;

procedure
TADODBTest.DataSetAfterEdit(DataSet: TDataSet);
begin
  LogEvent('AfterEdit', DataSet);
end;

procedure
TADODBTest.DataSetAfterDelete(DataSet: TDataSet);
begin
  LogEvent('AfterDelete', DataSet);
end;

procedure
TADODBTest.DataSetAfterCancel(
DataSet: TDataSet);
```

```
Connection);
end;

procedure
TADODBTest.ConnectionDisconnec
t(Connection: TADOConnection;
  var EventStatus: TEventStatus);
begin
  LogEvent('Disconnect',
Connection);
end;

procedure
TADODBTest.ConnectionExecuteC
omplete(Connection:
TADOConnection;
  RecordsAffected: Integer; const
Error: Error;
  var EventStatus: TEventStatus;
const Command: _Command;
  const Recordset: _Recordset);
begin
  LogEvent('ExecuteComplete',
Connection);
end;

procedure
TADODBTest.ConnectionInfoMessa
ge(Connection: TADOConnection;
  const Error: Error; var
EventStatus: TEventStatus);
begin
  LogEvent('InfoMessage',
Connection);
end;

procedure
TADODBTest.ConnectionRollbackT
ransComplete(
  Connection: TADOConnection;
const Error: Error;
  var EventStatus: TEventStatus);
begin

LogEvent('RollbackTransComplete',
Connection);
end;

procedure
```

```
  LogEvent(Format('FetchProgress:
%d of %d', [Progress,
MaxProgress]), DataSet);
end;

procedure
TADODBTest.FieldValidate(Sender:
TField);
begin
  LogEvent(Format('Val: %s=%s',
[Sender.DisplayName,
Sender.AsSTring]));
end;

{ Connection Events }

procedure
TADODBTest.ConnectionBeginTra
nsComplete(
  Connection: TADOConnection;
TransactionLevel: Integer;
  const Error: Error; var
EventStatus: TEventStatus);
begin
  LogEvent('BeginTransComplete',
Connection);
end;

procedure
TADODBTest.ConnectionCommitT
ransComplete(Connection:
TADOConnection;
  const Error: Error; var
EventStatus: TEventStatus);
begin

LogEvent('CommitTransComplete',
Connection);
end;

procedure
TADODBTest.ConnectionConnectC
omplete(Connection:
TADOConnection;
  const Error: Error; var
EventStatus: TEventStatus);
begin
  ClearProgressBar;
  LogEvent('ConnectComplete',
```

```
MasterQuery.Parameters else
   FParamSource :=
MasterProc.Parameters;
  if not Showing then Exit;
  UpdateParameterList;
end;

procedure
TADODBTest.RefreshParameters(Parameters: TParameters);
var
  I: Integer;
  NewParameter: TParameter;
begin
  try
    Parameters.Refresh;
  except
  end;
  if Parameters.Count = 0 then Exit;
  for I := 0 to Parameters.Count - 1
do
    with Parameters[I] do
      if Name[1] = '@' then
      begin
        NewParameter :=
Parameters.CreateParameter(Copy(Name, 2, 100), DataType, Direction,
Size, Null);
        NewParameter.Index := I;
        Parameters[I].Free;
      end;
end;

procedure
TADODBTest.ParamPageShow(Sender: TObject);
var
  FT: TFieldType;
begin
  if ParameterType.Items.Count = 0
then
    with ParameterType.Items do
      for FT := low(TFieldType) to
high(TFieldType) do
        Add(FieldTypeNames[FT]);
end;

procedure
TADODBTest.ConnectionWillConnect(Connection: TADOConnection;
  var ConnectionString, UserID,
Password: WideString;
  var ConnectOptions:
TConnectOption; var EventStatus:
TEventStatus);
begin
  LogEvent('WillConnect',
Connection);
end;

procedure
TADODBTest.ConnectionWillExecute(Connection: TADOConnection;
  var CommandText: WideString;
var CursorType: TCursorType;
  var LockType: TADOLockType;
var CommandType:
TCommandType;
  var ExecuteOptions:
TExecuteOptions; var EventStatus:
TEventStatus;
  const Command: _Command;
const Recordset: _Recordset);
begin
  LogEvent('WillExecute',
Connection);
end;

procedure
TADODBTest.ConnectionLogin(Sender: TObject; Username,
  Password: String);
begin
  LogEvent(Format('OnLogin - UID:
%s PWD: %s',[UserName,
Password]), Sender as
TADOConnection);
end;

{ Parameters }

procedure
TADODBTest.ParameterSourceClick(Sender: TObject);
begin
  if SQLParams.Checked then
    FParamSource :=
```

```
procedure TADODBTest.AddParameterButtonClick(Sender: TObject);
begin
  FParamSource.CreateParameter('Param'+IntToStr(FParamSource.Count+1), ftInteger, pdInput, 0, 0);
  FModifiedParameter := -1;
  UpdateParameterList;
end;

procedure TADODBTest.ParameterListClick(Sender: TObject);
begin
  WriteParameterData;
  if ParameterList.ItemIndex < 0 then Exit;
  with FParamSource[ParameterList.ItemIndex] do
  begin
    ParameterName.Text := Name;
    ParameterValue.Text := VarToStr(Value);
    ParameterType.Text := FieldTypeNames[DataType];
    ParameterSize.Text := IntToStr(Size);
    ParameterScale.Text := IntToStr(NumericScale);
    ParameterPrecision.Text := IntToStr(Precision);
    ParameterDirectionGroup.ItemIndex := Ord(Direction)-1;
    PANullableCheckbox.Checked := paNullable in Attributes;
    PALongCheckbox.Checked := paLong in Attributes;
    PASignedCheckbox.Checked := paSigned in Attributes;
  end;
  FModifiedParameter := -1;
end;

procedure TADODBTest.WriteParameterData;
```

```
TADODBTest.UpdateParameterList;
var
  I: Integer;
begin
  with ParameterList.Items do
  try
    BeginUpdate;
    Clear;
    for I := 0 to FParamSource.Count - 1 do
      Add(FParamSource[I].DisplayName);
    if ParameterList.Items.Count > 0 then
    begin
      if FModifiedParameter > -1 then
        ParameterList.ItemIndex := FModifiedParameter else
        ParameterList.ItemIndex := 0;
      ParameterListClick(ParameterList);
    end else
    begin
      ParameterName.Text := '';
      ParameterValue.Text := '';
    end;
  finally
    EndUpdate;
  end;
end;

procedure TADODBTest.RefreshParametersButtonClick(Sender: TObject);
begin
  CheckConnection(False);
  if SQLParams.Checked then
    MasterQuery.SQL.Text := MasterSQL.Text else
    MasterProc.ProcedureName := MasterProcName.Text;
  RefreshParameters(FParamSource);
  UpdateParameterList;
end;
```

```
var
  DataTypeIndex: Integer;
begin
  if FModifiedParameter < 0 then Exit;
  with FParamSource[FModifiedParameter] do
  begin
    if Name <> ParameterName.Text then
    begin
      Name := ParameterName.Text;
      ParameterList.Items[FModifiedParameter] := Name;
    end;
    DataTypeIndex := ParameterType.Items.IndexOf(ParameterType.Text);
    if DataTypeIndex <> -1 then
      DataType := TFieldtype(DataTypeIndex) else
      DataType := ftInteger;
    Size := StrToInt(ParameterSize.Text);
    NumericScale := StrToInt(ParameterScale.Text);
    Precision := StrToInt(ParameterPrecision.Text);
    Direction := TParameterDirection(ParameterDirectionGroup.ItemIndex+1);
    if PANullableCheckbox.Checked then
      Attributes := [paNullable];
    if PALongCheckbox.Checked then
      Attributes := Attributes + [paLong];
    if PASignedCheckbox.Checked then
      Attributes := Attributes + [paSigned];
    if VarToStr(Value) <> ParameterValue.Text then
      Value := ParameterValue.Text;
  end;
  FModifiedParameter := -1;
end;
```

```

procedure TADODBTest.ParameterDataChange(Sender: TObject);
begin
  FModifiedParameter := ParameterList.ItemIndex;
end;

procedure TADODBTest.MasterSQLChange(Sender: TObject);
begin
  ParameterList.Items.Clear;
end;

{ Test Code }

procedure TADODBTest.TestButtonClick(Sender: TObject);
begin
  { Put your test code here... }
end;


procedure TADODBTest.ToolBar1Click(Sender: TObject);
begin

end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

```

الملف النصي للبرنامج

```
object ADODBTest: TADODBTest
  Left = 175
  Top = 114
  HorzScrollBar.Increment = 52
  VertScrollBar.Increment = 45
  AutoScroll = False
  Caption = 'ADO DB Controls Test Application'
  ClientHeight = 460
  ClientWidth = 680
```

Anchors = [akLeft, akTop, akRight]
Constraints.MinWidth = 500
TabOrder = 1
object SourcePage: TTabSheet
Caption = 'Source'
object EditConnStr: TSpeedButton
Left = 460
Top = 14
Width = 18
Height = 18
Caption = '...'
Font.Charset = ANSI_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
OnClick = EditConnStrClick
end
object Label2: TLabel
Left = 6
Top = -1
Width = 87
Height = 13
Caption = 'Connection String:'
Font.Charset = ANSI_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
object TableNames: TGroupBox
Left = 5
Top = 38
Width = 133
Height = 76
Caption = ' Master/Detail Tables '
TabOrder = 1
object MasterTableName: TComboBox
Left = 8
Top = 18

Color = clBtnFace
ParentFont = True
Menu = MainMenu1
OldCreateOrder = True
Position = poScreenCenter
OnCloseQuery = FormCloseQuery
OnCreate = FormCreate
OnDestroy = FormDestroy
DesignSize = (
680
460)
PixelsPerInch = 96
TextHeight = 13
object Label1: TLabel
Left = 16
Top = 53
Width = 3
Height = 13
end
object StatusBar: TStatusBar
Left = 0
Top = 436
Width = 680
Height = 24
Panels = <
item
Text = 'Message'
Width = 300
end
item
Text = 'Modified'
Width = 80
end
item
Text = 'Null'
Width = 50
end
item
Text = 'RecNo'
Width = 50
end>
end
object AreaSelector: TPageControl
Left = 5
Top = 30
Width = 501
Height = 148
ActivePage = TabSheet1

```
      Action = PrevQuery
      Flat = True
      Glyph.Data = {
      ParentShowHint = False
      ShowHint = True
    end
    object NextQuery1: TSpeedButton
      Left = 189
      Top = 52
      Width = 19
      Height = 13
      Action = NextQuery
      Flat = True
      Glyph.Data = {
      ParentShowHint = False
      ShowHint = True
    end
    object EditCommandText: TSpeedButton
      Left = 189
      Top = 19
      Width = 18
      Height = 18
      Caption = '...'
      Font.Charset = ANSI_CHARSET
      Font.Color = clWindowText
      Font.Height = -13
      Font.Name = 'MS Sans Serif'
      Font.Style = [fsBold]
      ParentFont = False
      OnClick = EditCommandTextClick
    end
    object MasterSQL: TMemo
      Left = 7
      Top = 18
      Width = 180
      Height = 21
      Lines.Strings = (
        'Select * from Customer')
      TabOrder = 0
      OnChange = MasterSQLChange
      OnKeyPress = MasterSQLKeyPress
    end
```

```
      Width = 116
      Height = 21
      DropDownCount = 20
      ItemHeight = 13
      Sorted = True
      TabOrder = 0
      Text = 'Customer'
      OnClick = MasterTableNameClick
      OnDropDown = MasterTableNameDropDown
      OnKeyPress = MasterTableNameKeyPress
    end
    object DetailTableName: TComboBox
      Left = 8
      Top = 45
      Width = 116
      Height = 21
      DropDownCount = 20
      ItemHeight = 13
      Sorted = True
      TabOrder = 1
      Text = 'Orders'
      OnClick = DetailTableNameClick
      OnDropDown = MasterTableNameDropDown
      OnKeyPress = MasterTableNameKeyPress
    end
  end
  object QueryStrings: TGroupBox
    Left = 143
    Top = 38
    Width = 212
    Height = 75
    Caption = ' Master/Detail Queries '
    TabOrder = 2
    object PrevQuery1: TSpeedButton
      Left = 189
      Top = 39
      Width = 19
      Height = 13
```

MasterProcNameClick
OnDropDown = ProcNameDropDown
OnKeyPress = MasterProcNameKeyPress
end
object DetailProcName: TComboBox
Left = 9
Top = 44
Width = 108
Height = 21
ItemHeight = 13
TabOrder = 1
OnClick = DetailProcNameClick
OnDropDown = ProcNameDropDown
end
end
end
object FieldsPage: TTabSheet
Caption = 'Fields'
OnShow = FieldsPageShow
object FieldSchemaGrid: TDBGrid
Left = 0
Top = 0
Width = 493
Height = 120
Align = alClient
DataSource = FieldSchemaSource
TabOrder = 0
TitleFont.Charset = DEFAULT_CHARSET
TitleFont.Color = clWindowText
TitleFont.Height = -11
TitleFont.Name = 'MS Sans Serif'
TitleFont.Style = []
end
end
object ParamPage: TTabSheet
Caption = 'Parameters'
ImageIndex = 6
OnShow = ParamPageShow

object DetailSQL: TMemo
Left = 7
Top = 45
Width = 180
Height = 21
Lines.Strings = (
'select * from orders where CustNo = '
':CustNo')
TabOrder = 1
OnKeyPress = MasterSQLKeyPress
end
end
object ConnectionString: TComboBox
Left = 6
Top = 13
Width = 450
Height = 21
Hint = 'asdfasdf'
DropDownCount = 20
ItemHeight = 13
ParentShowHint = False
ShowHint = True
TabOrder = 0
OnClick = ConnectionStringClick
OnKeyPress = ConnectionStringKeyPress
end
object ProcedureNames: TGroupBox
Left = 361
Top = 38
Width = 125
Height = 75
Caption = ' Stored Procedure '
TabOrder = 3
object MasterProcName: TComboBox
Left = 9
Top = 17
Width = 108
Height = 21
ItemHeight = 13
TabOrder = 0
OnClick =

ItemHeight = 13
TabOrder = 0
OnClick = ParameterListClick
end
object ParameterName: TEdit
Left = 135
Top = 16
Width = 101
Height = 21
TabOrder = 1
OnChange = ParameterDataChange
end
object ParameterValue: TEdit
Left = 135
Top = 93
Width = 101
Height = 21
TabOrder = 2
OnChange = ParameterDataChange
end
object ParameterSize: TEdit
Left = 242
Top = 16
Width = 71
Height = 21
TabOrder = 3
OnChange = ParameterDataChange
end
object ParameterScale: TEdit
Left = 242
Top = 55
Width = 71
Height = 21
TabOrder = 4
OnChange = ParameterDataChange
end
object ParameterPrecision: TEdit
Left = 242
Top = 93
Width = 71
Height = 21
TabOrder = 5
OnChange =

object ParameterNameLabel: TLabel
Left = 135
Top = 2
Width = 28
Height = 13
Caption = 'Name'
end
object PTypeLabel: TLabel
Left = 135
Top = 40
Width = 24
Height = 13
Caption = 'Type'
end
object PValueLabel: TLabel
Left = 135
Top = 79
Width = 27
Height = 13
Caption = 'Value'
end
object PSizeLabel: TLabel
Left = 243
Top = 2
Width = 20
Height = 13
Caption = 'Size'
end
object PScaleLabel: TLabel
Left = 242
Top = 41
Width = 27
Height = 13
Caption = 'Scale'
end
object PPrecisionLabel: TLabel
Left = 242
Top = 79
Width = 43
Height = 13
Caption = 'Precision'
end
object ParameterList: TListBox
Left = 2
Top = 16
Width = 124
Height = 74

```
ParameterDataChange
      end
      object PALongCheckBox: TCheckBox
        Left = 87
        Top = 13
        Width = 52
        Height = 17
        Caption = 'Long'
        TabOrder = 2
        OnClick = ParameterDataChange
      end
    end
    object AddParameterButton: TButton
      Left = 7
      Top = 94
      Width = 54
      Height = 23
      Caption = 'Add'
      TabOrder = 8
      OnClick = AddParameterButtonClick
    end
    object RefreshParametersButton: TButton
      Left = 66
      Top = 94
      Width = 54
      Height = 23
      Caption = 'Refresh'
      TabOrder = 9
      OnClick = RefreshParametersButtonClick
    end
    object ParameterType: TComboBox
      Left = 135
      Top = 55
      Width = 101
      Height = 21
      ItemHeight = 13
      TabOrder = 10
      OnChange = ParameterDataChange
    end
    object SQLParams:
```

```
ParameterDataChange
    end
    object ParameterDirectionGroup: TRadioGroup
      Left = 322
      Top = 2
      Width = 162
      Height = 57
      Caption = ' Direction '
      Columns = 2
      Items.Strings = (
        'Input'
        'Output'
        'In/Out'
        'Return')
      TabOrder = 6
      OnClick = ParameterDataChange
    end
    object ParamAttributes: TGroupBox
      Left = 322
      Top = 63
      Width = 162
      Height = 52
      Caption = ' Attributes '
      TabOrder = 7
      object PANullableCheckBox: TCheckBox
        Left = 11
        Top = 13
        Width = 72
        Height = 17
        Caption = 'Nullable'
        TabOrder = 0
        OnClick = ParameterDataChange
      end
      object PASignedCheckBox: TCheckBox
        Left = 11
        Top = 31
        Width = 66
        Height = 17
        Caption = 'Signed'
        TabOrder = 1
        OnClick =
```

```
object Label13: TLabel
  Left = 156
  Top = 80
  Width = 68
  Height = 13
  Caption = 'CaseInsFields:'
end
object IndexList: TListBox
  Left = 8
  Top = 17
  Width = 137
  Height = 100
  ItemHeight = 13
  TabOrder = 0
  OnClick = IndexListClick
end
object IndexFields: TEdit
  Left = 156
  Top = 17
  Width = 164
  Height = 21
  ReadOnly = True
  TabOrder = 1
end
object DescFields: TEdit
  Left = 156
  Top = 56
  Width = 164
  Height = 21
  ReadOnly = True
  TabOrder = 2
end
object CaseInsFields: TEdit
  Left = 156
  Top = 96
  Width = 164
  Height = 21
  ReadOnly = True
  TabOrder = 3
end
object IndexOptions:
TGroupBox
  Left = 333
  Top = 2
  Width = 159
  Height = 113
  Caption = ' Index Options '
  TabOrder = 4
```

```
TRadioButton
    Left = 5
    Top = 1
    Width = 43
    Height = 14
    Caption = 'SQL'
    Checked = True
    TabOrder = 11
    TabStop = True
    OnClick =
ParameterSourceClick
  end
  object ProcParams:
TRadioButton
    Left = 50
    Top = 1
    Width = 73
    Height = 14
    Caption = 'StoredProc'
    TabOrder = 12
    OnClick =
ParameterSourceClick
  end
end
object IndexPage: TTabSheet
  Caption = 'Indexes'
  OnShow = IndexPageShow
  object Label11: TLabel
    Left = 10
    Top = 2
    Width = 65
    Height = 13
    Caption = 'Index Names:'
  end
  object Label14: TLabel
    Left = 156
    Top = 3
    Width = 30
    Height = 13
    Caption = 'Fields:'
  end
  object Label3: TLabel
    Left = 156
    Top = 40
    Width = 55
    Height = 13
    Caption = 'DescFields:'
  end
```

```
      Caption = 'Filter:'
    end
    object Filter: TEdit
      Left = 14
      Top = 17
      Width = 235
      Height = 21
      TabOrder = 0
      OnExit = FilterExit
      OnKeyPress = FilterKeyPress
    end
    object FindFirst: TButton
      Left = 174
      Top = 50
      Width = 75
      Height = 25
      Caption = 'F&ind First'
      TabOrder = 2
      OnClick = FindFirstClick
    end
    object FindNext: TButton
      Left = 174
      Top = 84
      Width = 75
      Height = 25
      Caption = 'Find &Next'
      TabOrder = 3
      OnClick = FindNextClick
    end
    object Filtered: TCheckBox
      Left = 28
      Top = 46
      Width = 97
      Height = 17
      Caption = 'Filtered'
      TabOrder = 1
      OnClick = FilteredClick
    end
    object FilterGroupBox: TRadioGroup
      Left = 288
      Top = 9
      Width = 110
      Height = 105
      Caption = ' Filter Group '
      Items.Strings = (
        'Pending'
        'Affected'
```

```
        object idxCaseInsensitive: TCheckBox
          Left = 12
          Top = 85
          Width = 109
          Height = 17
          Caption = 'Case Insensitive'
          Enabled = False
          TabOrder = 0
        end
        object idxDescending: TCheckBox
          Left = 12
          Top = 64
          Width = 86
          Height = 17
          Caption = 'Descending'
          Enabled = False
          TabOrder = 1
        end
        object idxPrimary: TCheckBox
          Left = 12
          Top = 21
          Width = 97
          Height = 17
          Caption = 'Primary'
          Enabled = False
          TabOrder = 2
        end
        object idxUnique: TCheckBox
          Left = 12
          Top = 42
          Width = 97
          Height = 17
          Caption = 'Unique'
          Enabled = False
          TabOrder = 3
        end
      end
    end
    object FilterPage: TTabSheet
      Caption = 'Filters'
      OnShow = FilterPageShow
      object Label10: TLabel
        Left = 14
        Top = 2
        Width = 25
        Height = 13
```

```
      end
      object LocateField: TComboBox
        Left = 10
        Top = 32
        Width = 183
        Height = 21
        ItemHeight = 13
        Sorted = True
        TabOrder = 0
        OnDropDown = LocateFieldDropDown
      end
      object locPartialKey: TCheckBox
        Left = 217
        Top = 74
        Width = 97
        Height = 16
        Caption = 'Partial Key'
        TabOrder = 4
      end
      object LocateNull: TCheckBox
        Left = 217
        Top = 48
        Width = 97
        Height = 17
        Caption = 'Null Value'
        TabOrder = 3
        OnClick = LocateNullClick
      end
    end
  end
  object TabSheet1: TTabSheet
    Caption = 'MiscTest'
    object TestButton: TButton
      Left = 8
      Top = 7
      Width = 75
      Height = 25
      Caption = 'Test'
      TabOrder = 0
      OnClick = TestButtonClick
    end
  end
end
object DataPanel: TPanel
  Left = 5
```

```
        'Fetched'
        'Conflicting')
      TabOrder = 4
      OnClick = FilterGroupBoxClick
    end
  end
  object LocatePage: TTabSheet
    Caption = 'Locate'
    OnShow = LocatePageShow
    object LocateGroupBox: TGroupBox
      Left = 4
      Top = 1
      Width = 329
      Height = 114
      Caption = ' Locate '
      TabOrder = 0
      object Label16: TLabel
        Left = 10
        Top = 16
        Width = 36
        Height = 13
        Caption = 'Field(s):'
      end
      object Label17: TLabel
        Left = 11
        Top = 61
        Width = 122
        Height = 13
        Caption = 'Values (comma delimited):'
      end
      object LocateButton: TButton
        Left = 216
        Top = 18
        Width = 75
        Height = 25
        Caption = '&Locate'
        TabOrder = 2
        OnClick = LocateButtonClick
      end
      object LocateEdit: TEdit
        Left = 10
        Top = 79
        Width = 183
        Height = 21
        TabOrder = 1
```

Checked = True
TabOrder = 1
TabStop = True
OnClick = ADOButtonClick
end
object MidasButton: TRadioButton
Left = 312
Top = 8
Width = 54
Height = 17
Caption = 'Midas'
TabOrder = 2
OnClick = MidasButtonClick
end
end
object BlobCtrlPanel: TPanel
Left = 315
Top = 30
Width = 185
Height = 219
Align = alRight
BevelOuter = bvNone
BorderWidth = 2
TabOrder = 2
Visible = False
object Splitter1: TSplitter
Left = 2
Top = 107
Width = 181
Height = 2
Cursor = crVSplit
Align = alBottom
Visible = False
end
object DBMemo1: TDBMemo
Left = 2
Top = 2
Width = 181
Height = 105
Align = alClient
DataSource = MasterDataSource
MaxLength = 1024
PopupMenu = PopupMenu1
TabOrder = 0
end
object DBImage1: TDBImage

Top = 183
Width = 501
Height = 250
Anchors = [akLeft, akTop, akRight, akBottom]
Constraints.MinHeight = 80
Constraints.MinWidth = 500
TabOrder = 2
object NavigatorPanel: TPanel
Left = 1
Top = 1
Width = 499
Height = 29
Align = alTop
BevelOuter = bvNone
TabOrder = 0
object ReadOnlyLabel: TLabel
Left = 403
Top = 9
Width = 60
Height = 13
Caption = 'Read Only'
Font.Charset = ANSI_CHARSET
Font.Color = clRed
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
Visible = False
end
object DBNavigator1: TDBNavigator
Left = 4
Top = 3
Width = 240
Height = 25
DataSource = MasterDataSource
TabOrder = 0
end
object ADOButton: TRadioButton
Left = 257
Top = 8
Width = 42
Height = 17
Caption = 'ADO'

```
      OnTitleClick = GridTitleClick
    end
    object MasterGrid: TDBGrid
      Left = 2
      Top = 2
      Width = 310
      Height = 105
      Align = alClient
      DataSource = MasterDataSource
      ParentShowHint = False
      ShowHint = True
      TabOrder = 1
      TitleFont.Charset = DEFAULT_CHARSET
      TitleFont.Color = clWindowText
      TitleFont.Height = -11
      TitleFont.Name = 'MS Sans Serif'
      TitleFont.Style = []
      OnColEnter = GridColEnter
      OnColumnMoved = MasterGridColumnMoved
      OnEnter = GridSetFocus
      OnTitleClick = GridTitleClick
    end
    object DBEditScroller: TScrollBox
      Left = 2
      Top = 2
      Width = 303
      Height = 0
      TabOrder = 2
    end
  end
end
object Events: TListBox
  Left = 512
  Top = 50
  Width = 164
  Height = 381
  Anchors = [akTop, akRight, akBottom]
  ItemHeight = 13
  TabOrder = 3
end
object ToolBar1: TToolBar
      Left = 2
      Top = 109
      Width = 181
      Height = 108
      Align = alBottom
      DataSource = MasterDataSource
      PopupMenu = PopupMenu1
      TabOrder = 1
    end
  end
  object GridPanel: TPanel
    Left = 1
    Top = 30
    Width = 314
    Height = 219
    Align = alClient
    BevelOuter = bvNone
    BorderWidth = 2
    TabOrder = 1
    object GridSplitter: TSplitter
      Left = 2
      Top = 107
      Width = 310
      Height = 2
      Cursor = crVSplit
      Align = alBottom
      Visible = False
    end
    object DetailGrid: TDBGrid
      Left = 2
      Top = 109
      Width = 310
      Height = 108
      Align = alBottom
      DataSource = DetailDataSource
      TabOrder = 0
      TitleFont.Charset = DEFAULT_CHARSET
      TitleFont.Color = clWindowText
      TitleFont.Height = -11
      TitleFont.Name = 'MS Sans Serif'
      TitleFont.Style = []
      Visible = False
      OnColEnter = GridColEnter
      OnEnter = GridSetFocus
```

```
  end
  object SavePacketButton: TToolButton
    Left = 129
    Top = 0
    Action = SaveToFile
  end
  object Sep2: TToolButton
    Left = 152
    Top = 0
    Width = 14
    Caption = 'Sep2'
    ImageIndex = 1
    Style = tbsDivider
  end
  object ClearFieldButton: TToolButton
    Left = 166
    Top = 0
    Action = ClearField
  end
  object Sep3: TToolButton
    Left = 189
    Top = 0
    Width = 14
    Caption = 'Sep3'
    ImageIndex = 1
    Style = tbsDivider
  end
  object BatchUpdateButton: TToolButton
    Left = 203
    Top = 0
    Action = BatchUpdate
  end
  object CancelBatchButton: TToolButton
    Left = 226
    Top = 0
    Action = CancelBatch
  end
  object Sep4: TToolButton
    Left = 249
    Top = 0
    Width = 14
    Caption = 'Sep4'
    ImageIndex = 21
    Style = tbsSeparator
```

```
  Left = 0
  Top = 0
  Width = 680
  Height = 29
  BorderWidth = 1
  Caption = 'ToolBar1'
  Flat = True
  Images = ImageList1
  ParentShowHint = False
  ShowHint = True
  TabOrder = 4
  OnClick = ToolBar1Click
  object OpenTableButton: TToolButton
    Left = 0
    Top = 0
    Action = OpenTable
  end
  object OpenQueryButton: TToolButton
    Left = 23
    Top = 0
    Action = OpenQuery
  end
  object ToolButton1: TToolButton
    Left = 46
    Top = 0
    Action = OpenProcedure
  end
  object CloseActiveButton: TToolButton
    Left = 69
    Top = 0
    Action = CloseActiveDataSet
  end
  object Sep1: TToolButton
    Left = 92
    Top = 0
    Width = 14
    Caption = 'Sep1'
    ImageIndex = 11
    Style = tbsSeparator
  end
  object LoadPacketButton: TToolButton
    Left = 106
    Top = 0
    Action = LoadFromFile
```

end
object OpenDialog: TOpenDialog
DefaultExt = 'dtg'
Filter =
'TableGram Files (*.dtg)|*.dtg|Delphi form (*.dfm)|*.dfm|All File' +
's (*.*)|*.*'
Title = 'Load Data File'
Left = 627
Top = 57
end
object SaveDialog: TSaveDialog
DefaultExt = 'dtg'
Filter =
'TableGram Files (*.dtg)|*.dtg|Delphi form (*.dfm)|*.dfm|All File' +
's (*.*)|*.*'
Options = [ofOverwritePrompt]
Title = 'Save Data File'
Left = 459
Top = 371
end
object PopupMenu1: TPopupMenu
OnPopup = PopupMenu1Popup
Left = 544
Top = 115
object AssignField1: TMenuItem
Caption = '&Assign Field'
OnClick = FieldSelect
end
end
object DetailDataSource: TDataSource
OnStateChange = DataSourceStateChange
OnDataChange = DataSourceDataChange
OnUpdateData = DataSourceUpdateData
Left = 19
Top = 393
end
object MainMenu1: TMainMenu
Left = 529
Top = 58
object FileMenu: TMenuItem

end
object MidasApplyUpdatesButton: TToolButton
Left = 263
Top = 0
Action = MidasApplyUpdates
end
object MidasCancelButton: TToolButton
Left = 286
Top = 0
Action = MidasCancelUpdates
end
object ToolButton3: TToolButton
Left = 309
Top = 0
Width = 14
Caption = 'ToolButton3'
ImageIndex = 9
Style = tbsSeparator
end
object ClearEventsButton: TToolButton
Left = 323
Top = 0
Action = ClearEventLog
end
end
object ProgressBar: TProgressBar
Left = 0
Top = 438
Width = 301
Height = 22
Step = 1
TabOrder = 5
Visible = False
end
object MasterDataSource: TDataSource
OnStateChange = DataSourceStateChange
OnDataChange = DataSourceDataChange
OnUpdateData = DataSourceUpdateData
Left = 19
Top = 283

end
object N1: TMenuItem
Caption = '-'
end
object Saveformtofile1: TMenuItem
Action = StreamFormOut
end
object LoadFormfromFile1: TMenuItem
Action = StreamFormIn
end
object N2: TMenuItem
Caption = '-'
end
object Exit1: TMenuItem
Action = ExitApplication
end
end
object EditMenu: TMenuItem
Caption = '&Edit'
object BatchUpdates1: TMenuItem
Action = BatchUpdate
end
object CancelBatch1: TMenuItem
Action = CancelBatch
end
object N6: TMenuItem
Caption = '-'
end
object ApplyUpdatesMidas1: TMenuItem
Action = MidasApplyUpdates
end
object CancelUpdatesMidas1: TMenuItem
Action = MidasCancelUpdates
end
object N7: TMenuItem
Caption = '-'
end
object ClearField1: TMenuItem
Action = ClearField
end
object RefreshData1: TMenuItem

Caption = '&File'
OnClick = FileMenuClick
object OpenTable1: TMenuItem
Action = OpenTable
end
object OpenQuery1: TMenuItem
Action = OpenQuery
end
object OpenProcedure1: TMenuItem
Action = OpenProcedure
end
object FileReopen: TMenuItem
Caption = 'Reopen'
object ReopenHolder: TMenuItem
Caption = 'PlaceHolder'
end
end
object ExecuteQuery1: TMenuItem
Action = ExecuteCommand
end
object CloseActiveDataset1: TMenuItem
Action = CloseActiveDataSet
end
object DisconnectDataSet1: TMenuItem
Action = DisconnectDataSet
end
object Disconnect1: TMenuItem
Action = CloseConnection
end
object N3: TMenuItem
Caption = '-'
end
object LoadDatapacket1: TMenuItem
Action = LoadFromFile
end
object SaveDataPacket1: TMenuItem
Action = SaveToFile
end
object LoadBlobfromfile1: TMenuItem
Action = LoadBlobFromFile

```
      object CurTypeForwardOnly: TMenuItem
        Caption = '&Forward Only'
        GroupIndex = 1
        RadioItem = True
        OnClick = RadioItemClick
      end
      object CurTypeKeyset: TMenuItem
        Caption = '&Keyset'
        Checked = True
        GroupIndex = 1
        RadioItem = True
        OnClick = RadioItemClick
      end
      object Dynamic1: TMenuItem
        Caption = '&Dynamic'
        GroupIndex = 1
        RadioItem = True
        OnClick = RadioItemClick
      end
      object CurTypeStatic: TMenuItem
        Caption = '&Static'
        GroupIndex = 1
        RadioItem = True
        OnClick = RadioItemClick
      end
    end
    object LockTypeItem: TMenuItem
      Caption = '&Lock Type'
      object LckTypeUnspecified: TMenuItem
        Caption = '&Unspecified'
        GroupIndex = 1
        RadioItem = True
        OnClick = RadioItemClick
      end
      object LckTypeReadOnly: TMenuItem
        Caption = '&Read Only'
        GroupIndex = 1
        RadioItem = True
        OnClick = RadioItemClick
      end
      object LckTypePessimistic: TMenuItem
```

```
      Action = RefreshData
    end
  end
  object ViewMenu: TMenuItem
    Caption = '&View'
    object DisplayDetailTable1: TMenuItem
      Action = DisplayDetails
    end
    object BlobfieldasImage1: TMenuItem
      Action = BlobAsImage
    end
    object N5: TMenuItem
      Caption = '-'
    end
    object ViewEvents1: TMenuItem
      Action = ViewEvents
    end
    object ClearEventLog1: TMenuItem
      Action = ClearEventLog
    end
  end
  object SettingsMenu: TMenuItem
    Caption = '&Settings'
    object UseClientCursorItem: TMenuItem
      Action = UseClientCursor
    end
    object UseadCmdTableDirect1: TMenuItem
      Action = UseTableDirect
    end
    object UseShapeProvider1: TMenuItem
      Action = UseShapeProvider
    end
    object CursorTypeItem: TMenuItem
      Caption = 'C&ursor Type'
      object CurTypeUnspecified: TMenuItem
        Caption = '&Unspecified'
        GroupIndex = 1
        RadioItem = True
        OnClick = RadioItemClick
      end
```

```
      Action = HelpAbout
      Caption = '&About...'
    end
  end
end
object ActionList1: TActionList
  Images = ImageList1
  Left = 624
  Top = 190
  object BatchUpdate: TAction
    Category = 'Edit'
    Caption = '&Apply Batch Updates'
    Hint = 'Apply pending batch updates to the server'
    ImageIndex = 3
    ShortCut = 16450
    OnExecute = BatchUpdateExecute
    OnUpdate = EditActionsUpdate
  end
  object CancelBatch: TAction
    Category = 'Edit'
    Caption = '&Cancel Batch Updates'
    Hint = 'Cancel pending batch updates'
    ImageIndex = 14
    OnExecute = CancelBatchExecute
  end
  object OpenProcedure: TAction
    Category = 'File'
    Caption = 'Open &Procedure'
    Hint = 'Open stored procedure(s)'
    ImageIndex = 12
    ShortCut = 16464
    OnExecute = OpenProcedureExecute
  end
  object OpenTable: TAction
    Category = 'File'
    Caption = 'Open &Table'
    Hint = 'Open Table(s)'
    ImageIndex = 0
    ShortCut = 16468
    OnExecute = OpenTableExecute
```

```
          Caption = '&Pessimistic'
          GroupIndex = 1
          RadioItem = True
          OnClick = RadioItemClick
        end
        object LckTypeOptimistic: TMenuItem
          Caption = '&Optimistic'
          Checked = True
          GroupIndex = 1
          RadioItem = True
          OnClick = RadioItemClick
        end
        object LckTypeBatchOptimistic: TMenuItem
          Caption = '&Batch Optimistic'
          GroupIndex = 1
          RadioItem = True
          OnClick = RadioItemClick
        end
      end
      object MaxRecords1: TMenuItem
        Action = MaxRecords
      end
      object EnableBCD1: TMenuItem
        Action = EnableBCD
      end
      object N4: TMenuItem
        Caption = '-'
      end
      object AsyncConnect1: TMenuItem
        Action = AsyncConnect
      end
      object AsyncExecute1: TMenuItem
        Action = AsyncExecute
      end
      object AsyncFetch1: TMenuItem
        Action = AsyncFetch
      end
    end
    object HelpMenu: TMenuItem
      Caption = '&Help'
      object HelpAboutItem: TMenuItem
```

```
    Category = 'File'
    Caption = '&Execute Command'
    Hint = 'Execute the current query'
    ShortCut = 16453
    OnExecute = ExecSQLExecute
  end
  object StreamFormOut: TAction
    Category = 'File'
    Caption = 'Stream &Form Out...'
    Hint = 'Stream for to a form file'
    OnExecute = StreamFormOutClick
  end
  object StreamFormIn: TAction
    Category = 'File'
    Caption = 'Stream Form &In...'
    Hint = 'Stream in form file'
    OnExecute = StreamFormInClick
  end
  object ClearField: TAction
    Category = 'Edit'
    Caption = '&Clear Field'
    Hint = 'Set the current field to Null'
    ImageIndex = 6
    ShortCut = 24643
    OnExecute = ClearFieldExecute
  end
  object ViewEvents: TAction
    Category = 'View'
    Caption = '&Events'
    Hint = 'Display event information'
    OnExecute = ViewEventsExecute
  end
  object DisplayDetails: TAction
    Category = 'View'
    Caption = '&Display Detail Table'
    Checked = True
    Hint = 'Display detail or nested table in the lower grid'
    OnExecute = DisplayDetailsExecute
  end
  object RefreshData: TAction
  end
  object OpenQuery: TAction
    Category = 'File'
    Caption = 'Open &Query'
    Hint = 'Open Query(s)'
    ImageIndex = 1
    ShortCut = 16465
    OnExecute = OpenQueryExecute
  end
  object LoadFromFile: TAction
    Category = 'File'
    Caption = 'Load data from a file...'
    Hint = 'Load data from a file'
    ImageIndex = 10
    ShortCut = 16460
    OnExecute = LoadFromFileExecute
  end
  object CloseActiveDataSet: TAction
    Category = 'File'
    Caption = '&Close Active Dataset'
    Hint = 'Close the active dataset'
    ImageIndex = 2
    OnExecute = CloseActiveDataSetExecute
    OnUpdate = FileActionsUpdate
  end
  object SaveToFile: TAction
    Category = 'File'
    Caption = '&Save data to a file...'
    Hint = 'Save data to a file'
    ImageIndex = 11
    ShortCut = 16467
    OnExecute = SaveToFileExecute
  end
  object ExitApplication: TAction
    Category = 'File'
    Caption = 'E&xit'
    Hint = 'Exit the application'
    ShortCut = 32856
    OnExecute = ExitApplicationExecute
  end
  object ExecuteCommand: TAction
```

OnExecute = BooleanActionExecute
end
object UseShapeProvider: TAction
Category = 'Settings'
Caption = 'Use &Shape Provider'
Hint = 'Use the Shape Provider'
OnExecute = UseShapeProviderExecute
end
object CloseConnection: TAction
Category = 'File'
Caption = '&Close Connection'
Hint = 'Close the current connection'
OnExecute = CloseConnectionExecute
end
object AsyncConnect: TAction
Category = 'Settings'
Caption = 'AsyncConnect'
Hint = 'Make the connection asynchronously'
OnExecute = BooleanActionExecute
end
object AsyncExecute: TAction
Category = 'Settings'
Caption = 'AsyncExecute'
Hint = 'Execute the command asynchronously'
OnExecute = BooleanActionExecute
end
object AsyncFetch: TAction
Category = 'Settings'
Caption = 'AsyncFetch'
Hint = 'Fetch records asynchronously'
OnExecute = BooleanActionExecute
end
object MaxRecords: TAction
Category = 'Settings'
Caption = '&Max Records...'
Hint = 'Specify the max number

Category = 'Edit'
Caption = 'Re&fresh Data'
OnExecute = RefreshDataExecute
end
object ClearEventLog: TAction
Category = 'View'
Caption = '&Clear Event Log'
Hint = 'Clear the event log'
ImageIndex = 7
OnExecute = ClearEventLogExecute
OnUpdate = ClearEventLogUpdate
end
object HelpAbout: TAction
Caption = 'HelpAbout'
Hint = 'Display About Box'
OnExecute = HelpAboutExecute
end
object PrevQuery: TAction
Category = 'File'
Hint = 'Display the previous query from the history list'
OnExecute = PrevQueryExecute
OnUpdate = PrevQueryUpdate
end
object NextQuery: TAction
Category = 'File'
Hint = 'Display the next query from the history list, or insert new one.'
OnExecute = NextQueryExecute
end
object UseClientCursor: TAction
Category = 'Settings'
Caption = 'Use &Client Cursor'
Checked = True
Hint = 'Use client side cursor'
OnExecute = BooleanActionExecute
end
object UseTableDirect: TAction
Category = 'Settings'
Caption = 'Use adCmd&TableDirect'
Hint = 'Use the adCmdTableDirect command type'

```
    Hint = 'View the current blob field in a DBImage'
    OnExecute = BlobAsImageExecute
    OnUpdate = BlobAsImageUpdate
  end
  object LoadBlobFromFile: TAction
    Category = 'Edit'
    Caption = '&Load Blob from file...'
    Hint = 'Load blob data from a file'
    OnExecute = LoadBlobFromFileExecute
  end
 end
 object ImageList1: TImageList
  Left = 347
  Top = 366
  Bitmap = {

    00000}
 end
 object Connection: TADOConnection
  LoginPrompt = False
  Provider =
    'C:\Program Files\Common Files\System\OLE DB\Data Links\DBDemos.u' +
    'dl'
  OnDisconnect = ConnectionDisconnect
  OnInfoMessage = ConnectionInfoMessage
  OnBeginTransComplete = ConnectionBeginTransComplete
  OnCommitTransComplete = ConnectionCommitTransComplete
  OnRollbackTransComplete = ConnectionRollbackTransComplete
  OnConnectComplete = ConnectionConnectComplete
  OnWillConnect = ConnectionWillConnect
  OnExecuteComplete =
```

```
of records in a dataset'
    OnExecute = MaxRecordsExecute
  end
  object MidasApplyUpdates: TAction
    Category = 'Edit'
    Caption = 'Apply Updates (Midas)'
    Hint = 'Apply pending Midas updates'
    ImageIndex = 13
    ShortCut = 16449
    OnExecute = MidasApplyUpdatesExecute
  end
  object MidasCancelUpdates: TAction
    Category = 'Edit'
    Caption = 'Cancel Updates (Midas)'
    Hint = 'Cancel pending Midas updates'
    ImageIndex = 9
    OnExecute = MidasCancelUpdatesExecute
  end
  object EnableBCD: TAction
    Category = 'Settings'
    Caption = 'EnableBCD'
    Checked = True
    Hint = 'Configure datasets to use BCD fields for numeric field types'
    OnExecute = BooleanActionExecute
  end
  object DisconnectDataSet: TAction
    Category = 'File'
    Caption = '&Disconnect DataSet'
    Hint = 'Disconnect the dataset fromt he connection'
    OnExecute = DisconnectDataSetExecute
  end
  object BlobAsImage: TAction
    Category = 'View'
    Caption = '&Blob field as Image'
```

BeforeInsert = DataSetBeforeInsert
AfterInsert = DataSetAfterInsert
BeforeEdit = DataSetBeforeEdit
AfterEdit = DataSetAfterEdit
BeforePost = DataSetBeforePost
AfterPost = DataSetAfterPost
BeforeCancel = DataSetBeforeCancel
AfterCancel = DataSetAfterCancel
BeforeDelete = DataSetBeforeDelete
AfterDelete = DataSetAfterDelete
BeforeScroll = DataSetBeforeScroll
AfterScroll = DataSetAfterScroll
OnDeleteError = DataSetError
OnEditError = DataSetError
OnNewRecord = DataSetNewRecord
OnPostError = DataSetError
Left = 435
Top = 252
end
object MasterQuery: TADOQuery
Connection = Connection
BeforeOpen = MasterQueryBeforeOpen
AfterOpen = MasterQueryAfterOpen
BeforeClose = MasterQueryBeforeClose
AfterClose = DataSetAfterClose
BeforeInsert = DataSetBeforeInsert
AfterInsert = DataSetAfterInsert
BeforeEdit = DataSetBeforeEdit
AfterEdit = DataSetAfterEdit
BeforePost = DataSetBeforePost
AfterPost = DataSetAfterPost
BeforeCancel = DataSetBeforeCancel
AfterCancel = DataSetAfterCancel
BeforeDelete = DataSetBeforeDelete
AfterDelete = DataSetAfterDelete

ConnectionExecuteComplete
OnWillExecute = ConnectionWillExecute
OnLogin = ConnectionLogin
Left = 541
Top = 162
end
object MasterTable: TADOTable
Connection = Connection
BeforeOpen = MasterTableBeforeOpen
AfterOpen = MasterTableAfterOpen
BeforeClose = MasterTableBeforeClose
AfterClose = DataSetAfterClose
BeforeInsert = DataSetBeforeInsert
AfterInsert = DataSetAfterInsert
BeforeEdit = DataSetBeforeEdit
AfterEdit = DataSetAfterEdit
BeforePost = DataSetBeforePost
AfterPost = DataSetAfterPost
BeforeCancel = DataSetBeforeCancel
AfterCancel = DataSetAfterCancel
BeforeDelete = DataSetBeforeDelete
AfterDelete = DataSetAfterDelete
BeforeScroll = DataSetBeforeScroll
AfterScroll = DataSetAfterScroll
OnDeleteError = DataSetError
OnEditError = DataSetError
OnNewRecord = DataSetNewRecord
OnPostError = DataSetError
Left = 540
Top = 355
end
object DetailTable: TADOTable
Connection = Connection
BeforeOpen = DetailTableBeforeOpen
AfterOpen = DataSetAfterOpen
BeforeClose = DataSetBeforeClose
AfterClose = DataSetAfterClose

```
    BeforeOpen = MasterProcBeforeOpen
    AfterOpen = MasterProcAfterOpen
    BeforeClose = DataSetBeforeClose
    AfterClose = DataSetAfterClose
    BeforeInsert = DataSetBeforeInsert
    AfterInsert = DataSetAfterInsert
    BeforeEdit = DataSetBeforeEdit
    AfterEdit = DataSetAfterEdit
    BeforePost = DataSetBeforePost
    AfterPost = DataSetAfterPost
    BeforeCancel = DataSetBeforeCancel
    AfterCancel = DataSetAfterCancel
    BeforeDelete = DataSetBeforeDelete
    AfterDelete = DataSetAfterDelete
    BeforeScroll = DataSetBeforeScroll
    AfterScroll = DataSetAfterScroll
    OnDeleteError = DataSetError
    OnEditError = DataSetError
    OnNewRecord = DataSetNewRecord
    OnPostError = DataSetError
    Parameters = <>
    Prepared = True
    Left = 543
    Top = 219
  end
  object ADODataSet: TADODataSet
    Connection = Connection
    BeforeOpen = DataSetBeforeOpen
    AfterOpen = DataSetAfterOpen
    BeforeClose = DataSetBeforeClose
    AfterClose = DataSetAfterClose
    BeforeInsert = DataSetBeforeInsert
    AfterInsert = DataSetAfterInsert
    BeforeEdit = DataSetBeforeEdit
    AfterEdit = DataSetAfterEdit
    BeforePost = DataSetBeforePost
    AfterPost = DataSetAfterPost
    BeforeCancel =
```

```
    BeforeScroll = DataSetBeforeScroll
    AfterScroll = DataSetAfterScroll
    OnDeleteError = DataSetError
    OnEditError = DataSetError
    OnNewRecord = DataSetNewRecord
    OnPostError = DataSetError
    Parameters = <>
    Left = 622
    Top = 123
  end
  object DetailQuery: TADOQuery
    Connection = Connection
    BeforeOpen = DetailQueryBeforeOpen
    AfterOpen = DataSetAfterOpen
    BeforeClose = DataSetBeforeClose
    BeforeInsert = DataSetBeforeInsert
    AfterInsert = DataSetAfterInsert
    BeforeEdit = DataSetBeforeEdit
    AfterEdit = DataSetAfterEdit
    BeforePost = DataSetBeforePost
    AfterPost = DataSetAfterPost
    BeforeCancel = DataSetBeforeCancel
    AfterCancel = DataSetAfterCancel
    BeforeDelete = DataSetBeforeDelete
    AfterDelete = DataSetAfterDelete
    BeforeScroll = DataSetBeforeScroll
    AfterScroll = DataSetAfterScroll
    OnDeleteError = DataSetError
    OnEditError = DataSetError
    OnNewRecord = DataSetNewRecord
    OnPostError = DataSetError
    DataSource = DetailQuerySource
    Parameters = <>
    Left = 189
    Top = 228
  end
  object MasterProc: TADOStoredProc
    Connection = Connection
```

end>
Left = 624
Top = 300
end
object Provider: TDataSetProvider
Left = 110
Top = 336
end
object DetailProc: TADOStoredProc
Connection = Connection
BeforeOpen = DetailProcBeforeOpen
AfterOpen = DataSetAfterOpen
BeforeClose = DataSetBeforeClose
AfterClose = DataSetAfterClose
BeforeInsert = DataSetBeforeInsert
AfterInsert = DataSetAfterInsert
BeforeEdit = DataSetBeforeEdit
AfterEdit = DataSetAfterEdit
BeforePost = DataSetBeforePost
AfterPost = DataSetAfterPost
BeforeCancel = DataSetBeforeCancel
AfterCancel = DataSetAfterCancel
BeforeDelete = DataSetBeforeDelete
AfterDelete = DataSetAfterDelete
BeforeScroll = DataSetBeforeScroll
AfterScroll = DataSetAfterScroll
OnDeleteError = DataSetError
OnEditError = DataSetError
OnNewRecord = DataSetNewRecord
OnPostError = DataSetError
DataSource = DetailProcSource
Parameters = <>
Prepared = True
Left = 360
Top = 252
end
object DetailProcSource: TDataSource
DataSet = MasterProc
Left = 265

DataSetBeforeCancel
AfterCancel = DataSetAfterCancel
BeforeDelete = DataSetBeforeDelete
AfterDelete = DataSetAfterDelete
BeforeScroll = DataSetBeforeScroll
AfterScroll = DataSetAfterScroll
OnDeleteError = DataSetError
OnEditError = DataSetError
OnNewRecord = DataSetNewRecord
OnPostError = DataSetError
Parameters = <>
Left = 632
Top = 243
end
object DetailMasterSource: TDataSource
DataSet = MasterTable
Left = 131
Top = 269
end
object DetailQuerySource: TDataSource
DataSet = MasterQuery
Left = 110
Top = 221
end
object MasterClientData: TClientDataSet
Aggregates = <>
Params = <>
ProviderName = 'Provider'
OnReconcileError = MasterClientDataReconcileError
Left = 228
Top = 336
end
object ADOCommand: TADOCommand
Connection = Connection
Prepared = True
Parameters = <
item
Size = -1
Value = Null

```
    Top = 253
  end
  object FieldSchemaSource: TDataSource
    DataSet = FieldSchema
    Left = 538
    Top = 287
  end
  object FieldSchema: TADODataSet
    Parameters = <>
    Left = 617
    Top = 366
    object FieldSchemaCOLUMN_NAME: TWideStringField
      DisplayLabel = 'FieldName'
      DisplayWidth = 20
      FieldName = 'COLUMN_NAME'
      Size = 30
    end
    object FieldSchemaDATA_TYPE: TWordField
      DisplayLabel = 'Type'
      DisplayWidth = 20
      FieldName = 'DATA_TYPE'
      OnGetText = FieldSchemaDATA_TYPEGetText
    end
    object FieldSchemaCHARACTER_MAXIMUM_LENGTH: TIntegerField
      DisplayLabel = 'Size'
      FieldName = 'CHARACTER_MAXIMUM_LENGTH'
    end
    object FieldSchemaNUMERIC_PRECISION: TWordField
      DisplayLabel = 'Precision'
      FieldName = 'NUMERIC_PRECISION'
    end
    object FieldSchemaNUMERIC_SCALE: TSmallintField
      DisplayLabel = 'Scale'
      FieldName = 'NUMERIC_SCALE'
    end
  end
  object DetailClientData: TClientDataSet
    Aggregates = <>
    Params = <>
    Left = 160
    Top = 384
  end
End
```

انتهى البرنامج والآن مع الصور والرسوم من إعداد علاء الدين اللباد ٠٩٤٤٥٧٥٣٧١

الملف التنفيذي لبرنامج

(Actions)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

Actions

ProcedureS 16

```
unit Main;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdActns, ExtActns, ActnList, ImgList, ComCtrls, ToolWin,
  StdCtrls, Menus, Buttons, ExtCtrls, jpeg, ListActns;

type
  TForm1 = class(TForm)
    ActionList1: TActionList;
    ImageList1: TImageList;
    RichEditBold1: TRichEditBold;
```

```
GroupedAction21: TMenuItem;
GroupedAction31: TMenuItem;
N1: TMenuItem;
AutoCheckAction1: TMenuItem;
FileRun1: TFileRun;
Label1: TLabel;
Run1: TMenuItem;
N2: TMenuItem;
FileOpen1: TFileOpen;
Open1: TMenuItem;
BrowseURL1: TBrowseURL;
DownLoadURL1: TDownLoadURL;
SendMail1: TSendMail;
GroupedAction12: TMenuItem;
BrowseURL2: TMenuItem;
DownloadURL2: TMenuItem;
SendMail2: TMenuItem;
ColorSelect1: TColorSelect;
Dialog1: TMenuItem;
SelectColor1: TMenuItem;
OpenPicture1: TOpenPicture;
FontEdit1: TFontEdit;
PreviousTab1: TPreviousTab;
NextTab1: TNextTab;
Button3: TBitBtn;
Button4: TBitBtn;
SelectFont1: TMenuItem;
OpenPicture2: TMenuItem;
PageControl2: TPageControl;
ListTab: TTabSheet;
AutoCheckTab: TTabSheet;
DialogTab: TTabSheet;
FormatTab: TTabSheet;
EditTab: TTabSheet;
SearchTab: TTabSheet;
ToolBar2: TToolBar;
ToolButton18: TToolButton;
ToolButton19: TToolButton;
ToolButton20: TToolButton;
ToolButton21: TToolButton;
Memo2: TMemo;
Image1: TImage;
Panel1: TPanel;
SpeedButton1: TSpeedButton;
Button2: TBitBtn;
Button1: TBitBtn;
ToolBar1: TToolBar;
RichEditItalic1: TRichEditItalic;
RichEditUnderline1: TRichEditUnderline;
RichEditStrikeOut1: TRichEditStrikeOut;
RichEditBullets1: TRichEditBullets;
RichEditAlignLeft1: TRichEditAlignLeft;
RichEditAlignRight1: TRichEditAlignRight;
RichEditAlignCenter1: TRichEditAlignCenter;
SearchFind1: TSearchFind;
SearchFindNext1: TSearchFindNext;
SearchReplace1: TSearchReplace;
SearchFindFirst1: TSearchFindFirst;
GroupAction1: TAction;
GroupAction2: TAction;
GroupAction3: TAction;
AutoCheckAction: TAction;
StatusBar1: TStatusBar;
MainMenu1: TMainMenu;
File1: TMenuItem;
FileExit1: TFileExit;
Exit1: TMenuItem;
Search1: TMenuItem;
Find1: TMenuItem;
FindFirst1: TMenuItem;
FindNext1: TMenuItem;
Replace1: TMenuItem;
Edit1: TMenuItem;
EditCut1: TEditCut;
EditCopy1: TEditCopy;
EditPaste1: TEditPaste;
EditSelectAll1: TEditSelectAll;
EditUndo1: TEditUndo;
EditDelete1: TEditDelete;
Cut1: TMenuItem;
Copy1: TMenuItem;
Paste1: TMenuItem;
Delete1: TMenuItem;
SelectAll1: TMenuItem;
Undo1: TMenuItem;
AutoCheck1: TMenuItem;
GroupedAction11: TMenuItem;
```

```
RadioButton2: TRadioButton;
RadioButton3: TRadioButton;
ToolButton16: TToolButton;
ToolButton17: TToolButton;
ToolButton22: TToolButton;
ToolButton23: TToolButton;
ListBox1: TListBox;
ListBox2: TListBox;
SpeedButton4: TSpeedButton;
SpeedButton5: TSpeedButton;
SpeedButton6: TSpeedButton;
SpeedButton7: TSpeedButton;
SpeedButton8: TSpeedButton;
ListControlCopySelection1:
TListControlCopySelection;
ListControlDeleteSelection1:
TListControlDeleteSelection;
ListControlSelectAll1:
TListControlSelectAll;
ListControlClearSelection1:
TListControlClearSelection;
ListControlMoveSelection1:
TListControlMoveSelection;
Label8: TLabel;
Label9: TLabel;
StaticText1: TStaticText;
StaticText2: TStaticText;
StaticText3: TStaticText;
Label11: TLabel;
Label7: TLabel;
Label10: TLabel;
ShortCutTab: TTabSheet;
Button7: TBitBtn;
ShortCutAction: TAction;
HotKey1: THotKey;
Button8: TBitBtn;
AddShortCut: TAction;
Label12: TLabel;
Label13: TLabel;
ShortCutList: TListBox;
Label14: TLabel;
FileOpenWith1: TFileOpenWith;
OpenWith1: TMenuItem;
FileTab: TTabSheet;
Label3: TLabel;
Button9: TBitBtn;
Button10: TBitBtn;
FileSaveAs1: TFileSaveAs;
ToolButton5: TToolButton;
ToolButton6: TToolButton;
ToolButton7: TToolButton;
ToolButton8: TToolButton;
ToolButton11: TToolButton;
ToolButton12: TToolButton;
ToolButton1: TToolButton;
ToolButton2: TToolButton;
RichEdit1: TRichEdit;
ToolBar3: TToolBar;
ToolButton3: TToolButton;
ToolButton4: TToolButton;
ToolButton9: TToolButton;
ToolButton10: TToolButton;
ToolButton13: TToolButton;
ToolButton14: TToolButton;
Memo3: TMemo;
ListBoxTab: TTabSheet;
CheckBox2: TCheckBox;
CheckBox1: TCheckBox;
StaticListAction1:
TStaticListAction;
DeleteAction: TAction;
AddAction: TAction;
ClearAction: TAction;
ActiveAction: TAction;
SetIndexAction: TAction;
Label4: TLabel;
Label5: TLabel;
Label6: TLabel;
ListView1: TListView;
ComboBoxEx1: TComboBoxEx;
AddBtn: TBitBtn;
AddEdit: TEdit;
DeleteBtn: TBitBtn;
DelEdit: TEdit;
Button5: TBitBtn;
IdxEdit: TEdit;
Button6: TBitBtn;
ActiveBtn: TBitBtn;
GroupedActions11: TMenuItem;
Previous1: TMenuItem;
Next1: TMenuItem;
ToolBar4: TToolBar;
ToolButton15: TToolButton;
SpeedButton2: TSpeedButton;
SpeedButton3: TSpeedButton;
RadioButton1: TRadioButton;
```

```
  { Private declarations }
<-------->    procedure UpdateShortCutList;
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.FileOpen1Accept(Sender: TObject);
begin
  Label3.Caption := FileOpen1.Dialog.FileName;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
  FontEdit1.Dialog.Font.Assign(Font);
  // Make sure the following actions are enabled even though they don't do anything
  GroupAction1.DisableIfNoHandler := False;
  GroupAction2.DisableIfNoHandler := False;
  GroupAction3.DisableIfNoHandler := False;
  AutoCheckAction.DisableIfNoHandler := False;
  UpdateShortCutList;
end;

procedure TForm1.ColorSelect1Accept(Sender: TObject);
begin
  Color := ColorSelect1.Dialog.Color;
```

```
    FilePrintSetup1: TFilePrintSetup;
    N3: TMenuItem;
    PrintSetup1: TMenuItem;
    Memo1: TMemo;
    Button11: TBitBtn;
    Button12: TBitBtn;
    LabeledEdit1: TLabeledEdit;
    Button13: TBitBtn;
<-------->    procedure FileOpen1Accept(Sender: TObject);
<-------->    procedure FormCreate(Sender: TObject);
<-------->    procedure ColorSelect1Accept(Sender: TObject);
<-------->    procedure OpenPicture1Accept(Sender: TObject);
<-------->    procedure FontEdit1Accept(Sender: TObject);
<-------->    procedure DeleteActionExecute(Sender: TObject);
<-------->    procedure AddActionExecute(Sender: TObject);
<-------->    procedure ClearActionExecute(Sender: TObject);
<-------->    procedure ActiveActionExecute(Sender: TObject);
<-------->    procedure SetIndexActionExecute(Sender: TObject);
<-------->    procedure ShortCutActionExecute(Sender: TObject);
<-------->    procedure AddShortCutExecute(Sender: TObject);
<-------->    procedure FileSaveAs1Accept(Sender: TObject);
<-------->    procedure SearchTabShow(Sender: TObject);
<-------->    procedure EditTabShow(Sender: TObject);
  private
```

```
StaticListAction1.Active;
  if StaticListAction1.Active then
    ActiveBtn.Caption := 'Set InActive'
  else
    ActiveBtn.Caption := 'Set Active';
end;

procedure TForm1.SetIndexActionExecute(Sender: TObject);
begin
  StaticListAction1.ItemIndex := StrToInt(IdxEdit.Text);
end;

procedure TForm1.ShortCutActionExecute(Sender: TObject);
begin
  ShowMessage('ShortCut action executed');
end;

procedure TForm1.UpdateShortCutList;
var
  I: Integer;
begin
  ShortCutList.Items.BeginUpdate;
  try
    ShortCutList.Items.Clear;
    ShortCutList.Items.Add(ShortCutToText(ShortCutAction.ShortCut));
    for I := 0 to ShortCutAction.SecondaryShortCuts.Count - 1 do
      ShortCutList.Items.Add(ShortCutToText(ShortCutAction.SecondaryShortCuts.ShortCuts[I]));
  finally
    ShortCutList.Items.EndUpdate;
  end;
end;

procedure
end;

procedure TForm1.OpenPicture1Accept(Sender: TObject);
begin
  Image1.Picture.LoadFromFile(OpenPicture1.Dialog.FileName);
end;

procedure TForm1.FontEdit1Accept(Sender: TObject);
begin
  Font.Assign(FontEdit1.Dialog.Font);
end;

procedure TForm1.DeleteActionExecute(Sender: TObject);
begin
  StaticListAction1.Items.Delete(StrToInt(DelEdit.Text));
end;

procedure TForm1.AddActionExecute(Sender: TObject);
begin
  with StaticListAction1.Items.Add do
    Caption := AddEdit.Text;
end;

procedure TForm1.ClearActionExecute(Sender: TObject);
begin
  StaticListAction1.Items.Clear;
end;

procedure TForm1.ActiveActionExecute(Sender: TObject);
begin
  StaticListAction1.Active := not
```

1-<---<--- (OBJECT) Form1: TForm1
Left = -4
Top = -4
BorderIcons = [biSystemMenu, biMinimize]
BorderStyle = bsSingle
Caption = 'Action Demo'
ClientHeight = 692
ClientWidth = 1024
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
Menu = MainMenu1
OldCreateOrder = False
ShowHint = True
OnCreate = FormCreate
DesignSize = (
1024
692)
PixelsPerInch = 96
TextHeight = 13
2- <---<--- (OBJECT) Label1: TLabel
Left = 6
Top = 658
Width = 197
Height = 13
Anchors = [akLeft, akBottom]
Caption = 'Below is a StatusBar with AutoHint = True'
end
<---<--- (OBJECT) Label8: TLabel
Left = 8
Top = 5
Width = 159
Height = 13
Caption = '(all of the menu items use actions)'
end
/////// (OBJECT) Label11: TLabel
Left = 7
Top = 321

TForm1.AddShortCutExecute(Sender: TObject);
begin
ShortCutAction.SecondaryShortCuts.Add(ShortCutToText(HotKey1.HotKey));
UpdateShortCutList;
end;
procedure TForm1.FileSaveAs1Accept(Sender: TObject);
begin
Memo1.Lines.SaveToFile(FileSaveAs1.Dialog.FileName);
end;
procedure TForm1.SearchTabShow(Sender: TObject);
begin
Memo2.SetFocus;
end;
procedure TForm1.EditTabShow(Sender: TObject);
begin
Memo3.SetFocus;
end;
end.

Actions
برنامج شامل بلغة البرمجة الدلفي
إعداد علاء الدين اللباد
٠٩٤٤٥٧٥٣٧١
(ACTION)
الملف النصي : ١٨٩ (OBJECT)

DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 2
end
/////// (OBJECT) PageControl2: TPageControl
Left = 5
Top = 26
Width = 470
Height = 290
ActivePage = ShortCutTab
TabOrder = 3
/////// (OBJECT) FileTab: TTabSheet
Caption = 'File'
ImageIndex = 8
/////// (OBJECT) Label3: TLabel
Left = 93
Top = 10
Width = 158
Height = 13
Caption = '(select File|Open and select a file)'
Font.Charset = DEFAULT_CHARSET
Font.Color = clRed
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
/////// (OBJECT) Button9: TBitBtn
Left = 6
Top = 5
Width = 75
Height = 25
Action = FileOpen1
Caption = '&Open...'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'

Width = 259
Height = 13
Caption = 'Any control with a Blue caption is connected to actions'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
/////// (OBJECT) StatusBar1: TStatusBar
Left = 0
Top = 673
Width = 1024
Height = 19
AutoHint = True
Panels = <>
end
/////// (OBJECT) Button3: TBitBtn
Left = 320
Top = 324
Width = 75
Height = 25
Action = PreviousTab1
Caption = '&Previous'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 1
end
/////// (OBJECT) Button4: TBitBtn
Left = 400
Top = 324
Width = 75
Height = 25
Action = NextTab1
Caption = '&Next'
Font.Charset =

```
      Glyph.Data = {
        000}
    end
    /////// (OBJECT ) Button12: TBitBtn
      Left = 134
      Top = 65
      Width = 75
      Height = 25
      Action = FileOpenWith1
      Caption = 'Open &With...'
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 4
    end
    /////// (OBJECT ) LabeledEdit1: TLabeledEdit
      Left = 219
      Top = 67
      Width = 161
      Height = 21
      EditLabel.Width = 156
      EditLabel.Height = 13
      EditLabel.Caption = 'Specify a filename or leave blank'
      TabOrder = 5
    end
    /////// (OBJECT ) Button13: TBitBtn
      Left = 7
      Top = 65
      Width = 75
      Height = 25
      Action = FileRun1
      Caption = '&Run...'
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 6
      Font.Style = []
      ParentFont = False
      TabOrder = 0
      Glyph.Data = {
        F7C}
    end
    /////// (OBJECT ) Button10: TBitBtn
      Left = 6
      Top = 35
      Width = 75
      Height = 25
      Action = FilePrintSetup1
      Caption = 'Print Set&up...'
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 1
    end
    /////// (OBJECT ) Memo1: TMemo
      Left = 89
      Top = 132
      Width = 366
      Height = 123
      Lines.Strings = (
        'Memo1')
      TabOrder = 2
    end
    /////// (OBJECT ) Button11: TBitBtn
      Left = 6
      Top = 132
      Width = 75
      Height = 25
      Action = FileSaveAs1
      Caption = 'Save &As...'
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 3
```

end
/////// (OBJECT) ToolButton14: TToolButton
Left = 285
Top = 2
Action = EditSelectAll1
end
end
/////// (OBJECT) Memo3: TMemo
Left = 0
Top = 57
Width = 462
Height = 205
Align = alClient
Lines.Strings = (
'Memo3')
TabOrder = 1
end
/////// (OBJECT) StaticText2: TStaticText
Left = 0
Top = 40
Width = 462
Height = 17
Align = alTop
Caption = '(all of these buttons use actions)'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 2
end
end
/////// (OBJECT) SearchTab: TTabSheet
Caption = 'Search'
ImageIndex = 5
OnShow = SearchTabShow
/////// (OBJECT) ToolBar2: TToolBar
Left = 0
Top = 0
Width = 462
Height = 40

end
end
/////// (OBJECT) EditTab: TTabSheet
Caption = 'Edit'
ImageIndex = 4
OnShow = EditTabShow
/////// (OBJECT) ToolBar3: TToolBar
Left = 0
Top = 0
Width = 462
Height = 40
ButtonHeight = 36
ButtonWidth = 57
Caption = 'ToolBar3'
Images = ImageList1
ShowCaptions = True
TabOrder = 0
/////// (OBJECT) ToolButton3: TToolButton
Left = 0
Top = 2
Action = EditCut1
end
/////// (OBJECT) ToolButton4: TToolButton
Left = 57
Top = 2
Action = EditCopy1
end
/////// (OBJECT) ToolButton9: TToolButton
Left = 114
Top = 2
Action = EditPaste1
end
/////// (OBJECT) ToolButton10: TToolButton
Left = 171
Top = 2
Action = EditDelete1
end
/////// (OBJECT) ToolButton13: TToolButton
Left = 228
Top = 2
Action = EditUndo1

/////// (OBJECT) StaticText3: TStaticText
Left = 0
Top = 40
Width = 462
Height = 17
Align = alTop
Caption = '(all of these buttons use actions)'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 2
end
end
/////// (OBJECT) AutoCheckTab: TTabSheet
Caption = 'AutoCheck'
ImageIndex = 1
/////// (OBJECT) SpeedButton2: TSpeedButton
Left = 241
Top = 77
Width = 101
Height = 22
Action = GroupAction2
AllowAllUp = True
GroupIndex = 1
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
/////// (OBJECT) SpeedButton3: TSpeedButton
Left = 242
Top = 51
Width = 100
Height = 22
Action = GroupAction1
AllowAllUp = True

ButtonHeight = 36
ButtonWidth = 58
Caption = 'ToolBar2'
Images = ImageList1
ShowCaptions = True
TabOrder = 0
/////// (OBJECT) ToolButton18: TToolButton
Left = 0
Top = 2
Action = SearchFind1
end
/////// (OBJECT) ToolButton19: TToolButton
Left = 58
Top = 2
Action = SearchFindNext1
end
/////// (OBJECT) ToolButton20: TToolButton
Left = 116
Top = 2
Action = SearchReplace1
end
/////// (OBJECT) ToolButton21: TToolButton
Left = 174
Top = 2
Action = SearchFindFirst1
end
end
/////// (OBJECT) Memo2: TMemo
Left = 0
Top = 57
Width = 462
Height = 205
Align = alClient
HideSelection = False
Lines.Strings = (
'Here is some sample text for testing the search and replace acti' +
'ons. Simply click one of the '
'actions to find or replace text within the memo.')
ScrollBars = ssVertical
TabOrder = 1
end

Font.Style = []
ParentFont = False
TabOrder = 0
end
/////// (OBJECT) CheckBox1: TCheckBox
Left = 25
Top = 174
Width = 124
Height = 17
Action = AutoCheckAction
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 1
end
/////// (OBJECT) ToolBar4: TToolBar
Left = 0
Top = 0
Width = 462
Height = 40
AutoSize = True
ButtonHeight = 36
ButtonWidth = 93
Caption = 'ToolBar4'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
Images = ImageList1
ParentFont = False
ShowCaptions = True
TabOrder = 2
/////// (OBJECT) ToolButton15: TToolButton
Left = 0
Top = 2
Action = AutoCheckAction
end
/////// (OBJECT) ToolButton16: TToolButton

GroupIndex = 1
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
/////// (OBJECT) Label9: TLabel
Left = 8
Top = 243
Width = 259
Height = 13
Caption = '(all of the controls on this tab are connected to actions)'
end
/////// (OBJECT) Label7: TLabel
Left = 9
Top = 51
Width = 170
Height = 112
AutoSize = False
Caption =
'The two checkboxes below are both connected to the same action a' +
'nd illustrate the use of the AutoCheck proeprty of actions. Als' +
'o notice the items on the AutoCheck menu stay in sync as well wi' +
'thout writting any code.'
WordWrap = True
end
/////// (OBJECT) CheckBox2: TCheckBox
Left = 25
Top = 152
Width = 124
Height = 17
Action = AutoCheckAction
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'

```
    /////// (OBJECT ) RadioButton2: TRadioButton
      Left = 240
      Top = 126
      Width = 113
      Height = 17
      Action = GroupAction2
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 4
    end
    /////// (OBJECT ) RadioButton3: TRadioButton
      Left = 240
      Top = 145
      Width = 113
      Height = 17
      Action = GroupAction3
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 5
    end
  end
  /////// (OBJECT ) ListTab: TTabSheet
    Caption = 'Static List'
    /////// (OBJECT ) Label4: TLabel
      Left = 14
      Top = 151
      Width = 63
      Height = 13
      Caption = 'ComboBoxEx'
    end
    /////// (OBJECT ) Label5: TLabel
      Left = 193
      Top = 145
      Width = 39
      Height = 13
```

```
        Left = 93
        Top = 2
        Width = 8
        Caption = 'ToolButton16'
        ImageIndex = 0
        Style = tbsSeparator
      end
      /////// (OBJECT ) ToolButton17: TToolButton
        Left = 101
        Top = 2
        Action = GroupAction1
        Grouped = True
        Style = tbsCheck
      end
      /////// (OBJECT ) ToolButton22: TToolButton
        Left = 194
        Top = 2
        Action = GroupAction2
        Grouped = True
        Style = tbsCheck
      end
      /////// (OBJECT ) ToolButton23: TToolButton
        Left = 287
        Top = 2
        Action = GroupAction3
        Grouped = True
        Style = tbsCheck
      end
    end
    /////// (OBJECT ) RadioButton1: TRadioButton
      Left = 240
      Top = 107
      Width = 113
      Height = 17
      Action = GroupAction1
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 3
    end
```

```
      Top = 10
      Width = 107
      Height = 25
      Action = AddAction
      Caption = '&Add Item'
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 2
    end
    /////// (OBJECT ) AddEdit: TEdit
      Left = 129
      Top = 12
      Width = 121
      Height = 21
      TabOrder = 3
      Text = 'New Item'
    end
    /////// (OBJECT ) DeleteBtn: TBitBtn
      Left = 15
      Top = 47
      Width = 107
      Height = 25
      Action = DeleteAction
      Caption = '&Delete Item at Index'
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 4
    end
    /////// (OBJECT ) DelEdit: TEdit
      Left = 129
      Top = 47
      Width = 121
      Height = 21
      TabOrder = 5
      Text = '0'
    end
```

```
      Caption = 'ListView'
    end
    /////// (OBJECT ) Label6: TLabel
      Left = 132
      Top = 112
      Width = 201
      Height = 29
      AutoSize = False
      Caption = 'The controls below are connected to same TStaticListAction'
      WordWrap = True
    end
    /////// (OBJECT ) ListView1: TListView
      Left = 190
      Top = 163
      Width = 143
      Height = 94
      Action = StaticListAction1
      Columns = <
        item
          Width = -2
          WidthType = (
            -2)
        end>
      HideSelection = False
      RowSelect = True
      SmallImages = ImageList1
      TabOrder = 0
      ViewStyle = vsReport
    end
    /////// (OBJECT ) ComboBoxEx1: TComboBoxEx
      Left = 16
      Top = 169
      Width = 128
      Height = 22
      Action = StaticListAction1
      ItemHeight = 16
      TabOrder = 1
      Text = 'StaticListAction1'
      Images = ImageList1
      DropDownCount = 8
    end
    /////// (OBJECT ) AddBtn: TBitBtn
      Left = 15
```

Action = ActiveAction
Caption = 'Set InActive'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 9
end
end
/////// (OBJECT) DialogTab: TTabSheet
Caption = 'Dialog'
ImageIndex = 2
/////// (OBJECT) Image1: TImage
Left = 0
Top = 28
Width = 462
Height = 234
Align = alClient
Proportional = True
end
/////// (OBJECT) Panel1: TPanel
Left = 0
Top = 0
Width = 462
Height = 28
Align = alTop
BevelOuter = bvNone
TabOrder = 0
/////// (OBJECT) SpeedButton1: TSpeedButton
Left = 2
Top = 2
Width = 83
Height = 22
Action = ColorSelect1
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
/////// (OBJECT) Button2:

/////// (OBJECT) Button5: TBitBtn
Left = 15
Top = 116
Width = 107
Height = 25
Action = ClearAction
Caption = '&Clear'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 6
end
/////// (OBJECT) IdxEdit: TEdit
Left = 129
Top = 83
Width = 121
Height = 21
TabOrder = 7
Text = '-1'
end
/////// (OBJECT) Button6: TBitBtn
Left = 15
Top = 82
Width = 107
Height = 25
Action = SetIndexAction
Caption = 'Set &Index'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 8
end
/////// (OBJECT) ActiveBtn: TBitBtn
Left = 267
Top = 10
Width = 75
Height = 25

Caption = 'ToolBar1'
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
Images = ImageList1
ParentFont = False
ShowCaptions = True
TabOrder = 0
/////// (OBJECT) ToolButton5: TToolButton
Left = 0
Top = 2
Action = RichEditBold1
end
/////// (OBJECT) ToolButton6: TToolButton
Left = 64
Top = 2
Action = RichEditItalic1
end
/////// (OBJECT) ToolButton7: TToolButton
Left = 128
Top = 2
Action = RichEditStrikeOut1
end
/////// (OBJECT) ToolButton8: TToolButton
Left = 192
Top = 2
Action = RichEditUnderline1
Wrap = True
end
/////// (OBJECT) ToolButton11: TToolButton
Left = 0
Top = 38
Action = RichEditBullets1
end
/////// (OBJECT) ToolButton12: TToolButton
Left = 64
Top = 38
Action = RichEditAlignLeft1
end

TBitBtn
Left = 93
Top = 2
Width = 83
Height = 22
Action = FontEdit1
Caption = 'Se&lect Font...'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 0
end
/////// (OBJECT) Button1: TBitBtn
Left = 183
Top = 2
Width = 87
Height = 22
Action = OpenPicture1
Caption = '&Open Picture...'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 1
end
end
end
/////// (OBJECT) FormatTab: TTabSheet
Caption = 'Format'
ImageIndex = 3
/////// (OBJECT) ToolBar1: TToolBar
Left = 0
Top = 0
Width = 462
Height = 76
AutoSize = True
ButtonHeight = 36
ButtonWidth = 64

ImageIndex = 7
/////// (OBJECT) SpeedButton4: TSpeedButton
Left = 164
Top = 21
Width = 90
Height = 19
Action = ListControlClearSelection1
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
/////// (OBJECT) SpeedButton5: TSpeedButton
Left = 164
Top = 44
Width = 90
Height = 19
Action = ListControlCopySelection1
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
/////// (OBJECT) SpeedButton6: TSpeedButton
Left = 164
Top = 68
Width = 90
Height = 19
Action = ListControlDeleteSelection1
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False

/////// (OBJECT) ToolButton1: TToolButton
Left = 128
Top = 38
Action = RichEditAlignCenter1
end
/////// (OBJECT) ToolButton2: TToolButton
Left = 192
Top = 38
Action = RichEditAlignRight1
end
end
/////// (OBJECT) RichEdit1: TRichEdit
Left = 0
Top = 93
Width = 462
Height = 169
Align = alClient
HideSelection = False
Lines.Strings = (
'RichEdit1')
TabOrder = 1
end
/////// (OBJECT) StaticText1: TStaticText
Left = 0
Top = 76
Width = 462
Height = 17
Align = alTop
Caption = '(all of these buttons use actions)'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 2
end
end
/////// (OBJECT) ListBoxTab: TTabSheet
Caption = 'List'

```
      Items.Strings = (
        'one'
        'two'
        'three'
        'four'
        'five'
        'six'
        'seven'
        'eight'
        'nine'
        'ten')
      MultiSelect = True
      TabOrder = 0
    end
    /////// (OBJECT ) ListBox2: TListBox
      Left = 260
      Top = 21
      Width = 154
      Height = 151
      ItemHeight = 13
      TabOrder = 1
    end
  end
  /////// (OBJECT ) ShortCutTab: TTabSheet
    Caption = 'ShortCuts'
    ImageIndex = 7
    /////// (OBJECT ) Label12: TLabel
      Left = 126
      Top = 41
      Width = 221
      Height = 13
      Caption = 'This action responds to the following shortcuts:'
    end
    /////// (OBJECT ) Label13: TLabel
      Left = 23
      Top = 139
      Width = 330
      Height = 30
      AutoSize = False
      Caption =
        'Type a shortcut key and click the Add ShortCut button to add a n' +
        'ew shortcut to the ShortCutAction'
    end
    /////// (OBJECT ) SpeedButton7: TSpeedButton
      Left = 164
      Top = 91
      Width = 90
      Height = 19
      Action = ListControlMoveSelection1
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
    end
    /////// (OBJECT ) SpeedButton8: TSpeedButton
      Left = 164
      Top = 115
      Width = 90
      Height = 19
      Action = ListControlSelectAll1
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
    end
    /////// (OBJECT ) Label10: TLabel
      Left = 20
      Top = 4
      Width = 298
      Height = 13
      Caption = 'These list actions work on listboxes, listviews and comboboxes.'
    end
    /////// (OBJECT ) ListBox1: TListBox
      Left = 18
      Top = 21
      Width = 138
      Height = 151
      ItemHeight = 13
```

Caption = 'Add ShortCut'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 2
end
/////// (OBJECT) ShortCutList: TListBox
Left = 126
Top = 58
Width = 121
Height = 69
ItemHeight = 13
TabOrder = 3
end
end
end
/////// (OBJECT) ActionList1: TActionList
Images = ImageList1
Left = 425
Top = 104
/////// (OBJECT) RichEditBold1: TRichEditBold
Category = 'Format'
AutoCheck = True
Caption = '&Bold'
Hint = 'Bold'
ImageIndex = 0
ShortCut = 16450
end
/////// (OBJECT) RichEditItalic1: TRichEditItalic
Category = 'Format'
AutoCheck = True
Caption = '&Italic'
Hint = 'Italic'
ImageIndex = 1
ShortCut = 16457
end
/////// (OBJECT) RichEditUnderline1: TRichEditUnderline
Category = 'Format'

FocusControl = HotKey1
WordWrap = True
end
/////// (OBJECT) Label14: TLabel
Left = 17
Top = 7
Width = 321
Height = 13
Caption =
'This tab illustrates the new SecondaryShortCuts property of TAct' +
'ion:'
end
/////// (OBJECT) Button7: TBitBtn
Left = 19
Top = 59
Width = 98
Height = 25
Action = ShortCutAction
Caption = 'ShortCut Action'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 0
end
/////// (OBJECT) HotKey1: THotKey
Left = 22
Top = 174
Width = 121
Height = 19
HotKey = 49217
Modifiers = [hkCtrl, hkAlt]
TabOrder = 1
end
/////// (OBJECT) Button8: TBitBtn
Left = 21
Top = 200
Width = 75
Height = 25
Action = AddShortCut

```
    AutoCheck = True
    Caption = '&Center'
    Hint = 'Center|Centers text between margins'
    ImageIndex = 7
  end
  /////// (OBJECT ) SearchFind1: TSearchFind
    Category = 'Search'
    Caption = '&Find...'
    Dialog.Options = [frDown, frFindNext, frHideUpDown]
    Hint = 'Find|Finds the specified text'
    ImageIndex = 8
    ShortCut = 16454
  end
  /////// (OBJECT ) SearchFindNext1: TSearchFindNext
    Category = 'Search'
    Caption = 'Find &Next'
    Hint = 'Find Next|Repeats the last find'
    ImageIndex = 9
    SearchFind = SearchFind1
    ShortCut = 114
  end
  /////// (OBJECT ) SearchReplace1: TSearchReplace
    Category = 'Search'
    Caption = '&Replace'
    Hint = 'Replace|Replaces specific text with different text'
    ImageIndex = 10
  end
  /////// (OBJECT ) SearchFindFirst1: TSearchFindFirst
    Category = 'Search'
    Caption = 'F&ind First'
    Hint = 'Find First|Finds the first occurance of specified text'
  end
  /////// (OBJECT ) GroupAction1: TAction
    Category = 'AutoCheck'
    AutoCheck = True
    Caption = 'Grouped Action 1'
    GroupIndex = 1
```

```
    AutoCheck = True
    Caption = '&Underline'
    Hint = 'Underline'
    ImageIndex = 2
    ShortCut = 16469
  end
  /////// (OBJECT ) RichEditStrikeOut1: TRichEditStrikeOut
    Category = 'Format'
    AutoCheck = True
    Caption = '&Strikeout'
    Hint = 'Strikeout'
    ImageIndex = 3
  end
  /////// (OBJECT ) RichEditBullets1: TRichEditBullets
    Category = 'Format'
    AutoCheck = True
    Caption = '&Bullets'
    Hint = 'Bullets|Inserts a bullet on the current line'
    ImageIndex = 4
  end
  /////// (OBJECT ) RichEditAlignLeft1: TRichEditAlignLeft
    Category = 'Format'
    AutoCheck = True
    Caption = 'Align &Left'
    Hint = 'Align Left|Aligns text at the left indent'
    ImageIndex = 5
  end
  /////// (OBJECT ) RichEditAlignRight1: TRichEditAlignRight
    Category = 'Format'
    AutoCheck = True
    Caption = 'Align &Right'
    Hint = 'Align Right|Aligns text at the right indent'
    ImageIndex = 6
  end
  /////// (OBJECT ) RichEditAlignCenter1: TRichEditAlignCenter
    Category = 'Format'
```

end
/////// (OBJECT) EditCut1: TEditCut
Category = 'Edit'
Caption = 'Cu&t'
Hint = 'Cut|Cuts the selection and puts it on the Clipboard'
ImageIndex = 12
ShortCut = 16472
end
/////// (OBJECT) EditCopy1: TEditCopy
Category = 'Edit'
Caption = '&Copy'
Hint = 'Copy|Copies the selection and puts it on the Clipboard'
ImageIndex = 13
ShortCut = 16451
end
/////// (OBJECT) EditPaste1: TEditPaste
Category = 'Edit'
Caption = '&Paste'
Hint = 'Paste|Inserts Clipboard contents'
ImageIndex = 14
ShortCut = 16470
end
/////// (OBJECT) EditSelectAll1: TEditSelectAll
Category = 'Edit'
Caption = 'Select &All'
Hint = 'Select All|Selects the entire document'
ShortCut = 16449
end
/////// (OBJECT) EditUndo1: TEditUndo
Category = 'Edit'
Caption = '&Undo'
Hint = 'Undo|Reverts the last action'
ImageIndex = 15
ShortCut = 16474
end
/////// (OBJECT) EditDelete1: TEditDelete
Category = 'Edit'

Hint =
'Grouped Action where only one action can be checked at any give ' +
'time'
end
/////// (OBJECT) GroupAction2: TAction
Category = 'AutoCheck'
AutoCheck = True
Caption = 'Grouped Action 2'
GroupIndex = 1
Hint =
'Grouped Action where only one action can be checked at any give ' +
'time'
end
/////// (OBJECT) GroupAction3: TAction
Category = 'AutoCheck'
AutoCheck = True
Caption = 'Grouped Action 3'
GroupIndex = 1
Hint =
'Grouped Action where only one action can be checked at any give ' +
'time'
end
/////// (OBJECT) AutoCheckAction: TAction
Category = 'AutoCheck'
AutoCheck = True
Caption = 'AutoCheck Action'
Hint =
'AutoCheck action: when it is executed it will automatically set ' +
'it''s checked state without having to write any code'
end
/////// (OBJECT) FileExit1: TFileExit
Category = 'File'
Caption = 'E&xit'
Hint = 'Exit|Quits the application'
ImageIndex = 11

item
Caption = 'five'
ImageIndex = 5
end
item
Caption = 'six'
end>
end
/////// (OBJECT) DeleteAction: TAction
Category = 'Buttons'
Caption = '&Delete Item at Index'
Hint = 'Delete the item indicated from the StaticListAction'
OnExecute = DeleteActionExecute
end
/////// (OBJECT) AddAction: TAction
Category = 'Buttons'
Caption = '&Add Item'
Hint = 'Add the item to the StaticListAction'
OnExecute = AddActionExecute
end
/////// (OBJECT) ClearAction: TAction
Category = 'Buttons'
Caption = '&Clear'
Hint = 'Clear the StaticListAction'
OnExecute = ClearActionExecute
end
/////// (OBJECT) ActiveAction: TAction
Category = 'Buttons'
Caption = 'Set InActive'
Hint = 'Toggle the Active property of the StaticListAction'
OnExecute = ActiveActionExecute
end
/////// (OBJECT) SetIndexAction: TAction
Category = 'Buttons'
Caption = 'Set &Index'

Caption = '&Delete'
Hint = 'Delete|Erases the selection'
ImageIndex = 16
ShortCut = 46
end
/////// (OBJECT) FileRun1: TFileRun
Category = 'File'
Browse = True
BrowseDlg.Filter = 'Programs|*.exe;*.com;*.bat;*.cmd|All Files|*.*'
BrowseDlg.Options = [ofHideReadOnly, ofFileMustExist, ofEnableSizing]
BrowseDlg.Title = 'Run'
Caption = '&Run...'
Hint = 'Run|Runs an application'
Operation = 'open'
ShowCmd = scShowNormal
end
/////// (OBJECT) StaticListAction1: TStaticListAction
Category = 'List'
Caption = 'StaticListAction1'
Images = ImageList1
Items = <
item
Caption = 'zero'
ImageIndex = 0
end
item
Caption = 'one'
ImageIndex = 1
end
item
Caption = 'two'
ImageIndex = 2
end
item
Caption = 'three'
ImageIndex = 3
end
item
Caption = 'four'
ImageIndex = 4
end

Category = 'Dialog'
Caption = '&Open Picture...'
Hint = 'Open Picture'
ShortCut = 16463
OnAccept = OpenPicture1Accept
end
/////// (OBJECT) FontEdit1: TFontEdit
Category = 'Dialog'
Caption = 'Se&lect Font...'
Dialog.Font.Charset = DEFAULT_CHARSET
Dialog.Font.Color = clWindowText
Dialog.Font.Height = -11
Dialog.Font.Name = 'MS Sans Serif'
Dialog.Font.Style = []
Hint = 'Font Select'
OnAccept = FontEdit1Accept
end
/////// (OBJECT) PreviousTab1: TPreviousTab
Category = 'Tab'
TabControl = PageControl2
Caption = '&Previous'
Enabled = False
Hint = 'Previous|Go back to the previous tab'
end
/////// (OBJECT) NextTab1: TNextTab
Category = 'Tab'
TabControl = PageControl2
Caption = '&Next'
Enabled = False
Hint = 'Next|Go to the next tab'
end
/////// (OBJECT) ListControlCopySelection1: TListControlCopySelection
Category = 'List'
Caption = 'Copy >>'
Destination = ListBox2
ListControl = ListBox1
end
/////// (OBJECT) ListControlDeleteSelection1:

Hint = 'Set the selected item of the StaticListAction'
OnExecute = SetIndexActionExecute
end
/////// (OBJECT) FileOpen1: TFileOpen
Category = 'File'
Caption = '&Open...'
Hint = 'Open|Opens an existing file'
ImageIndex = 17
ShortCut = 16463
OnAccept = FileOpen1Accept
end
/////// (OBJECT) BrowseURL1: TBrowseURL
Category = 'Internet'
Caption = '&Browse URL'
Hint = 'Browses to http://www.borland.com'
URL = 'http://www.borland.com'
end
/////// (OBJECT) DownLoadURL1: TDownLoadURL
Category = 'Internet'
Caption = '&Download URL'
Hint = 'Downloads http://www.borland.com to a file called ''test'''
URL = 'http://www.borland.com'
Filename = 'test'
end
/////// (OBJECT) SendMail1: TSendMail
Category = 'Internet'
Caption = '&Send Mail...'
Hint = 'Send email'
end
/////// (OBJECT) ColorSelect1: TColorSelect
Category = 'Dialog'
Caption = 'Select &Color...'
Hint = 'Color Select'
OnAccept = ColorSelect1Accept
end
/////// (OBJECT) OpenPicture1: TOpenPicture

```
    Caption = 'Open &With...'
    Dialog.Filter = 'All Files|*.*'
    Dialog.Options = 
[ofHideReadOnly, ofFileMustExist, 
ofEnableSizing]
    Dialog.Title = 'Open With'
  end
  /////// (OBJECT ) FileSaveAs1: 
TFileSaveAs
    Category = 'File'
    Caption = 'Save &As...'
    Dialog.Filter = 'Text File|*.txt|All 
Files|*.*'
    Hint = 'Save As|Saves the active 
file with a new name'
    ImageIndex = 18
    OnAccept = FileSaveAs1Accept
  end
  /////// (OBJECT ) FilePrintSetup1: 
TFilePrintSetup
    Category = 'File'
    Caption = 'Print Set&up...'
    Hint = 'Print Setup'
  end
 end
 /////// (OBJECT ) ImageList1: 
TImageList
  Left = 426
  Top = 150
  Bitmap = {

    000000000000}
 end
 /////// (OBJECT ) MainMenu1: 
TMainMenu
  Images = ImageList1
  Left = 424
  Top = 58
  /////// (OBJECT ) File1: 
TMenuItem
    Caption = '&File'
    /////// (OBJECT ) Open1: 
TMenuItem
      Action = FileOpen1
    end
    /////// (OBJECT ) OpenWith1: 
TMenuItem
      Action = FileOpenWith1
```

```
TListControlDeleteSelection
    Category = 'List'
    Caption = 'Delete >>'
    ListControl = ListBox2
  end
  /////// (OBJECT ) 
ListControlSelectAll1: 
TListControlSelectAll
    Category = 'List'
    Caption = '<< Select All'
  end
  /////// (OBJECT ) 
ListControlClearSelection1: 
TListControlClearSelection
    Category = 'List'
    Caption = 'Clear'
  end
  /////// (OBJECT ) 
ListControlMoveSelection1: 
TListControlMoveSelection
    Category = 'List'
    Caption = 'Move >>'
    Destination = ListBox2
    ListControl = ListBox1
  end
  /////// (OBJECT ) ShortCutAction: 
TAction
    Category = 'ShortCut'
    Caption = 'ShortCut Action'
    ShortCut = 16466
    SecondaryShortCuts.Strings = (
      'Ctrl+Alt+W'
      'Ctrl+Alt+X'
      'Ctrl+Alt+Y'
      'Ctrl+Shift+Z')
    OnExecute = 
ShortCutActionExecute
  end
  /////// (OBJECT ) AddShortCut: 
TAction
    Category = 'ShortCut'
    Caption = 'Add ShortCut'
    OnExecute = 
AddShortCutExecute
  end
  /////// (OBJECT ) FileOpenWith1: 
TFileOpenWith
    Category = 'File'
```

////// (OBJECT) Search1: TMenuItem
Caption = '&Search'
////// (OBJECT) Find1: TMenuItem
Action = SearchFind1
end
////// (OBJECT) FindFirst1: TMenuItem
Action = SearchFindFirst1
end
////// (OBJECT) FindNext1: TMenuItem
Action = SearchFindNext1
end
////// (OBJECT) Replace1: TMenuItem
Action = SearchReplace1
end
end
////// (OBJECT) AutoCheck1: TMenuItem
Caption = '&AutoCheck'
////// (OBJECT) GroupedAction11: TMenuItem
Action = GroupAction1
AutoCheck = True
end
////// (OBJECT) GroupedAction21: TMenuItem
Action = GroupAction2
AutoCheck = True
end
////// (OBJECT) GroupedAction31: TMenuItem
Action = GroupAction3
AutoCheck = True
end
////// (OBJECT) N1: TMenuItem
Caption = '-'
end
////// (OBJECT) AutoCheckAction1: TMenuItem
Action = AutoCheckAction
AutoCheck = True
end
end
////// (OBJECT)

end
////// (OBJECT) N3: TMenuItem
Caption = '-'
end
////// (OBJECT) PrintSetup1: TMenuItem
Action = FilePrintSetup1
end
////// (OBJECT) Run1: TMenuItem
Action = FileRun1
end
////// (OBJECT) N2: TMenuItem
Caption = '-'
end
////// (OBJECT) Exit1: TMenuItem
Action = FileExit1
end
end
////// (OBJECT) Edit1: TMenuItem
Caption = '&Edit'
////// (OBJECT) Cut1: TMenuItem
Action = EditCut1
end
////// (OBJECT) Copy1: TMenuItem
Action = EditCopy1
end
////// (OBJECT) Paste1: TMenuItem
Action = EditPaste1
end
////// (OBJECT) Delete1: TMenuItem
Action = EditDelete1
end
////// (OBJECT) SelectAll1: TMenuItem
Action = EditSelectAll1
end
////// (OBJECT) Undo1: TMenuItem
Action = EditUndo1
end
end

```
GroupedAction12: TMenuItem
    Caption = '&Internet'
   /////// (OBJECT ) BrowseURL2: TMenuItem
     Action = BrowseURL1
    end
   /////// (OBJECT ) DownloadURL2: TMenuItem
     Action = DownLoadURL1
    end
   /////// (OBJECT ) SendMail2: TMenuItem
     Action = SendMail1
    end
  end
  /////// (OBJECT ) Dialog1: TMenuItem
    Caption = 'Dialog'
   /////// (OBJECT ) SelectColor1: TMenuItem
     Action = ColorSelect1
    end
   /////// (OBJECT ) SelectFont1: TMenuItem
     Action = FontEdit1
    end
   /////// (OBJECT ) OpenPicture2: TMenuItem
     Action = OpenPicture1
    end
  end
  /////// (OBJECT ) GroupedActions11: TMenuItem
    AutoCheck = True
    Caption = 'Tab'
    Checked = True
   /////// (OBJECT ) Previous1: TMenuItem
     Action = PreviousTab1
    end
   /////// (OBJECT ) Next1: TMenuItem
     Action = NextTab1
    end
  end
 end
end[1]
```

هذا البرنامج يحتوي على أكثر من ١٠٠ تعليمة وموضوعا مختلفا وهو برنامج شامل لكل التعليمات البرمجية السابقة التي مرت خلال الدروس السابقة وعند قدرة الطالب على تشكيل هذا البرنامج بنفسه يصبح قادرا على وضع نفسه في أول طريق البرمجة وعندها نقول له مبروك فان طريق الألف ميل يبدأ بخطوة وأنت قد قطعت أكثر من ربع المسافة لكي تصبح مبرمجا وتكون قادرا على تشكيل أي برنامج بنفسك حسب المعطيات التي توصلت إليها وبهذا البرنامج تكون قد صنعت على الأقل ٢٠ برنامجا فرعيا وأنا أؤمن بان من يحفظ ألف بيت شعر يستطيع أن يكتب قصيدة وبالتالي فان من يحفظ ٢٠ برنامجا مختلفا يستطيع أن يكون مبرمجا .

علاء الدين اللباد
٠٩٤٤٥٧٥٣٧١

ALAEDDIN LUBBAD C.C.

[1] علاء الدين اللباد للمحاسبة والبرمجيات ٠٩٤٤٥٧٥٣٧١

```
Form1);
  Application.Run;
end.
```

الواجهة الرئيسية (الملف التنفيذي)

```
unit Main;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdActns, ExtActns, ActnList, ImgList, ComCtrls, ToolWin,
  StdCtrls, Menus, Buttons, ExtCtrls, jpeg, ListActns;

type
  TForm1 = class(TForm)
    ActionList1: TActionList;
    ImageList1: TImageList;
    RichEditBold1: TRichEditBold;
    RichEditItalic1: TRichEditItalic;
    RichEditUnderline1: TRichEditUnderline;
    RichEditStrikeOut1: TRichEditStrikeOut;
    RichEditBullets1: TRichEditBullets;
    RichEditAlignLeft1: TRichEditAlignLeft;
    RichEditAlignRight1: TRichEditAlignRight;
    RichEditAlignCenter1: TRichEditAlignCenter;
    SearchFind1: TSearchFind;
    SearchFindNext1: TSearchFindNext;
    SearchReplace1: TSearchReplace;
    SearchFindFirst1: TSearchFindFirst;
    GroupAction1: TAction;
    GroupAction2: TAction;
    GroupAction3: TAction;
    AutoCheckAction: TAction;
    StatusBar1: TStatusBar;
    MainMenu1: TMainMenu;
```

الملف التنفيذي للبرنامج (program Actions) ويحتوي على ٣٢ (procedure) مختلفا بحيث يتعلم الطالب أكثر التعليمات البرمجية المتداولة في لغة البرمجة الدلفي ٧

```
program Actions;

Uses
  Forms,
  Main in '..\..\ÇãËáÉ ãä ÇáÈÑäÇãÌ ÏáÝí 7\Actions\Main.pas' {Form1};

{$R *.res}

Begin
  Application.Initialize;
  Application.CreateForm(TForm1,
```

Button3: TBitBtn;
Button4: TBitBtn;
SelectFont1: TMenuItem;
OpenPicture2: TMenuItem;
PageControl2: TPageControl;
ListTab: TTabSheet;
AutoCheckTab: TTabSheet;
DialogTab: TTabSheet;
FormatTab: TTabSheet;
EditTab: TTabSheet;
SearchTab: TTabSheet;
ToolBar2: TToolBar;
ToolButton18: TToolButton;
ToolButton19: TToolButton;
ToolButton20: TToolButton;
ToolButton21: TToolButton;
Memo2: TMemo;
Image1: TImage;
Panel1: TPanel;
SpeedButton1: TSpeedButton;
Button2: TBitBtn;
Button1: TBitBtn;
ToolBar1: TToolBar;
ToolButton5: TToolButton;
ToolButton6: TToolButton;
ToolButton7: TToolButton;
ToolButton8: TToolButton;
ToolButton11: TToolButton;
ToolButton12: TToolButton;
ToolButton1: TToolButton;
ToolButton2: TToolButton;
RichEdit1: TRichEdit;
ToolBar3: TToolBar;
ToolButton3: TToolButton;
ToolButton4: TToolButton;
ToolButton9: TToolButton;
ToolButton10: TToolButton;
ToolButton13: TToolButton;
ToolButton14: TToolButton;
Memo3: TMemo;
ListBoxTab: TTabSheet;
CheckBox2: TCheckBox;
CheckBox1: TCheckBox;
StaticListAction1: TStaticListAction;
DeleteAction: TAction;
AddAction: TAction;
ClearAction: TAction;

File1: TMenuItem;
FileExit1: TFileExit;
Exit1: TMenuItem;
Search1: TMenuItem;
Find1: TMenuItem;
FindFirst1: TMenuItem;
FindNext1: TMenuItem;
Replace1: TMenuItem;
Edit1: TMenuItem;
EditCut1: TEditCut;
EditCopy1: TEditCopy;
EditPaste1: TEditPaste;
EditSelectAll1: TEditSelectAll;
EditUndo1: TEditUndo;
EditDelete1: TEditDelete;
Cut1: TMenuItem;
Copy1: TMenuItem;
Paste1: TMenuItem;
Delete1: TMenuItem;
SelectAll1: TMenuItem;
Undo1: TMenuItem;
AutoCheck1: TMenuItem;
GroupedAction11: TMenuItem;
GroupedAction21: TMenuItem;
GroupedAction31: TMenuItem;
N1: TMenuItem;
AutoCheckAction1: TMenuItem;
FileRun1: TFileRun;
Label1: TLabel;
Run1: TMenuItem;
N2: TMenuItem;
FileOpen1: TFileOpen;
Open1: TMenuItem;
BrowseURL1: TBrowseURL;
DownLoadURL1: TDownLoadURL;
SendMail1: TSendMail;
GroupedAction12: TMenuItem;
BrowseURL2: TMenuItem;
DownloadURL2: TMenuItem;
SendMail2: TMenuItem;
ColorSelect1: TColorSelect;
Dialog1: TMenuItem;
SelectColor1: TMenuItem;
OpenPicture1: TOpenPicture;
FontEdit1: TFontEdit;
PreviousTab1: TPreviousTab;
NextTab1: TNextTab;

StaticText1: TStaticText;
StaticText2: TStaticText;
StaticText3: TStaticText;
Label11: TLabel;
Label7: TLabel;
Label10: TLabel;
ShortCutTab: TTabSheet;
Button7: TBitBtn;
ShortCutAction: TAction;
HotKey1: THotKey;
Button8: TBitBtn;
AddShortCut: TAction;
Label12: TLabel;
Label13: TLabel;
ShortCutList: TListBox;
Label14: TLabel;
FileOpenWith1: TFileOpenWith;
OpenWith1: TMenuItem;
FileTab: TTabSheet;
Label3: TLabel;
Button9: TBitBtn;
Button10: TBitBtn;
FileSaveAs1: TFileSaveAs;
FilePrintSetup1: TFilePrintSetup;
N3: TMenuItem;
PrintSetup1: TMenuItem;
Memo1: TMemo;
Button11: TBitBtn;
Button12: TBitBtn;
LabeledEdit1: TLabeledEdit;
Button13: TBitBtn;
ALALUBBAD1: TMenuItem;
ACC1: TMenuItem;
Action1: TAction;
Window1: TMenuItem;
Show1: TMenuItem;
Hide1: TMenuItem;
N4: TMenuItem;
ArrangeAll1: TMenuItem;
Cascade1: TMenuItem;
ile1: TMenuItem;
NewWindow1: TMenuItem;
(---------procedure-------) FileOpen1Accept(Sender: TObject);
(---------procedure-------) FormCreate(Sender: TObject);
(---------procedure-------) ColorSelect1Accept(Sender:

ActiveAction: TAction;
SetIndexAction: TAction;
Label4: TLabel;
Label5: TLabel;
Label6: TLabel;
ListView1: TListView;
ComboBoxEx1: TComboBoxEx;
AddBtn: TBitBtn;
AddEdit: TEdit;
DeleteBtn: TBitBtn;
DelEdit: TEdit;
Button5: TBitBtn;
IdxEdit: TEdit;
Button6: TBitBtn;
ActiveBtn: TBitBtn;
GroupedActions11: TMenuItem;
Previous1: TMenuItem;
Next1: TMenuItem;
ToolBar4: TToolBar;
ToolButton15: TToolButton;
SpeedButton2: TSpeedButton;
SpeedButton3: TSpeedButton;
RadioButton1: TRadioButton;
RadioButton2: TRadioButton;
RadioButton3: TRadioButton;
ToolButton16: TToolButton;
ToolButton17: TToolButton;
ToolButton22: TToolButton;
ToolButton23: TToolButton;
ListBox1: TListBox;
ListBox2: TListBox;
SpeedButton4: TSpeedButton;
SpeedButton5: TSpeedButton;
SpeedButton6: TSpeedButton;
SpeedButton7: TSpeedButton;
SpeedButton8: TSpeedButton;
ListControlCopySelection1: TListControlCopySelection;
ListControlDeleteSelection1: TListControlDeleteSelection;
ListControlSelectAll1: TListControlSelectAll;
ListControlClearSelection1: TListControlClearSelection;
ListControlMoveSelection1: TListControlMoveSelection;
Label8: TLabel;
Label9: TLabel;

(---------procedure-------) TForm1.FileOpen1Accept(Sender: TObject);
begin
Label3.Caption := FileOpen1.Dialog.FileName;
end;
(---------procedure-------) TForm1.FormCreate(Sender: TObject);
begin
FontEdit1.Dialog.Font.Assign(Font);
// Make sure the following actions are enabled even though they don't do anything
GroupAction1.DisableIfNoHandler := False;
GroupAction2.DisableIfNoHandler := False;
GroupAction3.DisableIfNoHandler := False;
AutoCheckAction.DisableIfNoHandler := False;
UpdateShortCutList;
end;
(---------procedure-------) TForm1.ColorSelect1Accept(Sender : TObject);
begin
Color := ColorSelect1.Dialog.Color;
end;
(---------procedure-------) TForm1.OpenPicture1Accept(Sender: TObject);
begin
Image1.Picture.LoadFromFile(OpenPicture1.Dialog.FileName);
end;
(---------procedure-------) TForm1.FontEdit1Accept(Sender: TObject);

TObject);
(---------procedure-------) OpenPicture1Accept(Sender: TObject);
(---------procedure-------) FontEdit1Accept(Sender: TObject);
(---------procedure-------) DeleteActionExecute(Sender: TObject);
(---------procedure-------) AddActionExecute(Sender: TObject);
(---------procedure-------) ClearActionExecute(Sender: TObject);
(---------procedure-------) ActiveActionExecute(Sender: TObject);
(---------procedure-------) SetIndexActionExecute(Sender: TObject);
(---------procedure-------) ShortCutActionExecute(Sender: TObject);
(---------procedure-------) AddShortCutExecute(Sender: TObject);
(---------procedure-------) FileSaveAs1Accept(Sender: TObject);
(---------procedure-------) SearchTabShow(Sender: TObject);
(---------procedure-------) EditTabShow(Sender: TObject);
private
{ Private declarations }
(---------procedure-------) UpdateShortCutList;
public
{ Public declarations }
end;
var
Form1: TForm1;
implementation
{$R *.dfm}

```
end;

 (---------procedure-------)
TForm1.ShortCutActionExecute(Sender: TObject);
begin
  ShowMessage('ShortCut action executed');
end;

 (---------procedure-------)
TForm1.UpdateShortCutList;
var
  I: Integer;
begin
  ShortCutList.Items.BeginUpdate;
  try
    ShortCutList.Items.Clear;

ShortCutList.Items.Add(ShortCutToText(ShortCutAction.ShortCut));
    for I := 0 to ShortCutAction.SecondaryShortCuts.Count - 1 do

ShortCutList.Items.Add(ShortCutToText(ShortCutAction.SecondaryShortCuts.ShortCuts[I]));
  finally
    ShortCutList.Items.EndUpdate;
  end;
end;

 (---------procedure-------)
TForm1.AddShortCutExecute(Sender: TObject);
begin

ShortCutAction.SecondaryShortCuts.Add(ShortCutToText(HotKey1.HotKey));
  UpdateShortCutList;
end;

 (---------procedure-------)
TForm1.FileSaveAs1Accept(Sender: TObject);
begin

begin

Font.Assign(FontEdit1.Dialog.Font);
end;

 (---------procedure-------)
TForm1.DeleteActionExecute(Sender: TObject);
begin

StaticListAction1.Items.Delete(StrToInt(DelEdit.Text));
end;

 (---------procedure-------)
TForm1.AddActionExecute(Sender: TObject);
begin
  with StaticListAction1.Items.Add do
    Caption := AddEdit.Text;
end;

 (---------procedure-------)
TForm1.ClearActionExecute(Sender: TObject);
begin
  StaticListAction1.Items.Clear;
end;

 (---------procedure-------)
TForm1.ActiveActionExecute(Sender: TObject);
begin
  StaticListAction1.Active := not StaticListAction1.Active;
  if StaticListAction1.Active then
    ActiveBtn.Caption := 'Set InActive'
  else
    ActiveBtn.Caption := 'Set Active';
end;

 (---------procedure-------)
TForm1.SetIndexActionExecute(Sender: TObject);
begin
  StaticListAction1.ItemIndex := StrToInt(IdxEdit.Text);
```

Memo1.Lines.SaveToFile(FileSaveAs1.Dialog.FileName);
end;
(---------procedure-------) TForm1.SearchTabShow(Sender: TObject);
begin
Memo2.SetFocus;
end;
(---------procedure-------) TForm1.EditTabShow(Sender: TObject);
begin
Memo3.SetFocus;
end;
end.

انتهى البرنامج ونأمل أن تكونوا قد حققتم الفائدة المرجوة منه والله ولي التوفيق.
علاء الدين اللباد
٠٩٤٤٥٧٥٣٧١

Caption = 'Marine Adventures'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -16
Font.Name = 'Times New Roman'
Font.Style = [fsItalic, fsUnderline]
ParentFont = False
end
-----< object \\ Bevel1: TBevel
Left = 10
Top = 150
Width = 147
Height = 2
Style = bsRaised
end
-----< object \\ Label1: TLabel
Left = 60
Top = 158
Width = 47
Height = 13
Caption = 'Loading...'
end
-----< object \\ Image1: TImage
Left = 40
Top = 11
Width = 87
Height = 113
Picture.Data = {
end
end
End
-----< object \\ SearchDlg: TSearchDlg
Left = 226
Top = 118
ActiveControl = DBGrid1
BorderIcons = [biSystemMenu]
BorderStyle = bsDialog
Caption = 'SearchDlg'
ClientHeight = 299
ClientWidth = 282
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET

<-------> Objects 10
الملف النصي لبرنامج
(MastApp)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

الملف النصي للبرنامج اعداد
علاء الدين اللباد
٠٩٤٤٥٧٥٣٧١
160 object
-----< object \\ SplashForm: TSplashForm
Left = 605
Top = 209
ActiveControl = Panel1
BorderIcons = []
BorderStyle = bsNone
ClientHeight = 178
ClientWidth = 168
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
PixelsPerInch = 96
TextHeight = 13
-----< object \\ Panel1: TPanel
Left = 0
Top = 0
Width = 168
Height = 178
Align = alClient
BevelInner = bvLowered
Color = clSilver
TabOrder = 0
-----< object \\ Label3: TLabel
Left = 23
Top = 125
Width = 116
Height = 19

dgRowSelect]
TabOrder = 2
TitleFont.Charset = DEFAULT_CHARSET
TitleFont.Color = clBlack
TitleFont.Height = -11
TitleFont.Name = 'MS Sans Serif'
TitleFont.Style = []
OnDblClick = DBGrid1DblClick
Columns = <
item
Expanded = False
FieldName = 'PartNo'
Visible = True
end
item
Expanded = False
FieldName = 'Description'
Width = 64
Visible = True
end
item
Expanded = False
FieldName = 'VendorNo'
Width = 64
Visible = True
end
item
Expanded = False
FieldName = 'OnHand'
Width = 64
Visible = True
end
item
Expanded = False
FieldName = 'OnOrder'
Visible = True
end
item
Expanded = False
FieldName = 'BackOrd'
Width = 64
Visible = True
end
item
Expanded = False
FieldName = 'Cost'
Width = 64

Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
PixelsPerInch = 96
TextHeight = 13
-----< object \\ Label1: TLabel
Left = 9
Top = 11
Width = 69
Height = 13
Alignment = taRightJustify
AutoSize = False
Caption = 'Search Field:'
end
-----< object \\ Label2: TLabel
Left = 7
Top = 40
Width = 71
Height = 13
Alignment = taRightJustify
AutoSize = False
Caption = 'Search Text:'
end
-----< object \\ SearchButton: TSpeedButton
Left = 248
Top = 36
Width = 23
Height = 23
Hint = 'Search'
Enabled = False
Glyph.Data = {
NumGlyphs = 2
ParentShowHint = False
ShowHint = True
OnClick = SearchButtonClick
end
-----< object \\ DBGrid1: TDBGrid
Left = 8
Top = 66
Width = 264
Height = 181
DataSource = DataSource
Options = [dgTitles, dgIndicator, dgColLines, dgRowLines,

-----< object \\ DataSource: TDataSource
DataSet = MastData.Parts
Left = 47
Top = 253
end
End
\\
-----< object \\ QueryCustDlg: TQueryCustDlg
Left = 372
Top = 297
ActiveControl = FromEdit
BorderIcons = [biSystemMenu]
BorderStyle = bsDialog
Caption = 'Specify Date Range'
ClientHeight = 96
ClientWidth = 318
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
-----< object \\ Bevel1: TBevel
Left = 5
Top = 6
Width = 225
Height = 84
Shape = bsFrame
end
-----< object \\ Label1: TLabel
Left = 12
Top = 34
Width = 23
Height = 13
Caption = '&From'
FocusControl = FromEdit
end
-----< object \\ Label2: TLabel

Visible = True
end
item
Expanded = False
FieldName = 'ListPrice'
Visible = True
end>
end
-----< object \\ OKBtn: TButton
Left = 115
Top = 265
Width = 75
Height = 25
Caption = '&OK'
Default = True
ModalResult = 1
TabOrder = 3
end
-----< object \\ CancelBtn: TButton
Left = 199
Top = 265
Width = 75
Height = 25
Cancel = True
Caption = '&Cancel'
ModalResult = 2
TabOrder = 4
end
-----< object \\ SearchEd: TEdit
Left = 80
Top = 36
Width = 161
Height = 21
TabOrder = 1
OnChange = SearchEdChange
OnKeyPress = SearchEdKeyPress
end
-----< object \\ OrderCombo: TComboBox
Left = 80
Top = 8
Width = 192
Height = 21
Style = csDropDownList
ItemHeight = 13
TabOrder = 0
OnChange = OrderComboChange
end

```
  ShowHint = True
  OnClick = PopupCalToBtnClick
 end
 -----< object \\ FromEdit: TEdit
  Left = 46
  Top = 31
  Width = 147
  Height = 21
  TabOrder = 0
 end
 -----< object \\ ToEdit: TEdit
  Left = 46
  Top = 60
  Width = 147
  Height = 21
  TabOrder = 1
 end
 -----< object \\ CancelBtn: TButton
  Left = 240
  Top = 34
  Width = 70
  Height = 25
  Cancel = True
  Caption = '&Cancel'
  ModalResult = 2
  TabOrder = 2
 end
 -----< object \\ OkBtn: TButton
  Left = 240
  Top = 6
  Width = 70
  Height = 25
  Caption = '&OK'
  Default = True
  ModalResult = 1
  TabOrder = 3
  OnClick = OkBtnClick
 end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\\\\\\\
-----< object \\ PickRpt: TPickRpt
 Left = 535
 Top = 233
 HelpContext = 5
 BorderStyle = bsDialog
 Caption = 'Report Selection'
 ClientHeight = 106
```

```
  Left = 12
  Top = 63
  Width = 13
  Height = 13
  Caption = '&To'
  FocusControl = ToEdit
 end
 -----< object \\ Msglab: TLabel
  Left = 12
  Top = 11
  Width = 197
  Height = 16
  AutoSize = False
  Caption = 'Customers with
LastInvoiceDate ranging:'
  WordWrap = True
 end
 -----< object \\ PopupCalBtnFrom:
TSpeedButton
  Left = 197
  Top = 31
  Width = 21
  Height = 21
  Hint = 'Browse calendar'
  Glyph.Data = {


3333330000003333333333333333330
00000}
  Layout = blGlyphRight
  ParentShowHint = False
  ShowHint = True
  OnClick =
PopupCalBtnFromClick
 end
 -----< object \\ PopupCalToBtn:
TSpeedButton
  Left = 197
  Top = 60
  Width = 21
  Height = 21
  Hint = 'Browse calendar'
  Glyph.Data = {


3333330000003333333333333333330
00000}
  Layout = blGlyphRight
  ParentShowHint = False
```

Left = 151
Top = 40
Width = 73
Height = 25
Caption = '&View'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ModalResult = 1
ParentFont = False
TabOrder = 2
OnClick = ViewBtnClick
end
-----< object \\ ReportType: TRadioGroup
Left = 9
Top = 6
Width = 129
Height = 87
Caption = '&Report'
ItemIndex = 0
Items.Strings = (
'C&ustomers Report'
'&Orders Report'
'&Invoice Report')
TabOrder = 3
end
End
\\
-----< object \\ PickOrderNoDlg: TPickOrderNoDlg
Left = 380
Top = 170
AutoScroll = False
Caption = 'Order No.'
ClientHeight = 190
ClientWidth = 166
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -11
Font.Name = 'MS Sans Serif'

ClientWidth = 238
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
PixelsPerInch = 96
TextHeight = 13
-----< object \\ OKBtn: TButton
Left = 150
Top = 11
Width = 73
Height = 25
Caption = '&Print'
Default = True
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ModalResult = 1
ParentFont = False
TabOrder = 0
OnClick = OKBtnClick
end
-----< object \\ CloseBtn: TButton
Left = 151
Top = 69
Width = 73
Height = 25
Cancel = True
Caption = '&Cancel'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ModalResult = 2
ParentFont = False
TabOrder = 1
end
-----< object \\ ViewBtn: TButton

\\\ \\\
-----< object \\ OrdersByDateReport: TOrdersByDateReport
Left = 0
Top = 0
Width = 816
Height = 780
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'Arial'
Font.Style = []
OldCreateOrder = True
PixelsPerInch = 96
TextHeight = 16
End
\\\
-----< object \\ MainForm: TMainForm
Left = 224
Top = 122
HelpContext = 1
ActiveControl = MainPanel
BorderIcons = [biSystemMenu, biMinimize]
BorderStyle = bsSingle
Caption = 'Marine Adventures Order Entry'
ClientHeight = 84
ClientWidth = 401
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
Menu = MainMenu
OldCreateOrder = True
OnCreate = FormCreate
OnDestroy = FormDestroy
PixelsPerInch = 96

Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
OnShow = FormShow
PixelsPerInch = 96
TextHeight = 13
-----< object \\ Label1: TLabel
Left = 5
Top = 1
Width = 94
Height = 13
Caption = '&Select an Order No.'
FocusControl = DBLookupListBox1
end
-----< object \\ CancelBtn: TButton
Left = 86
Top = 160
Width = 75
Height = 25
Cancel = True
Caption = '&Cancel'
ModalResult = 2
TabOrder = 0
end
-----< object \\ OKBtn: TButton
Left = 2
Top = 160
Width = 75
Height = 25
Caption = '&OK'
Default = True
ModalResult = 1
TabOrder = 1
end
-----< object \\ DBLookupListBox1: TDBLookupListBox
Left = 5
Top = 16
Width = 153
Height = 134
KeyField = 'OrderNo'
ListField = 'OrderNo'
ListSource = MastData.OrdersSource
TabOrder = 2
end
End

Caption = 'P&arts'
Glyph.Data = {
Layout = blGlyphTop
Spacing = 3
OnClick = BrowseParts
end
-----< object \\ ReportBtn: TSpeedButton
Left = 196
Top = 1
Width = 66
Height = 45
Hint = 'Print custom reports'
Caption = '&Reports'
Glyph.Data = {
0000}
Layout = blGlyphTop
Spacing = 0
OnClick = ReportBtnClick
end
-----< object \\ HelpBtn: TSpeedButton
Left = 261
Top = 1
Width = 66
Height = 45
Hint = 'View technical help about the application'
Caption = '&Help'
Glyph.Data = {
0303}
Layout = blGlyphTop
NumGlyphs = 2
Spacing = 0
OnClick = HelpBtnClick
end
-----< object \\ CloseBtn: TSpeedButton
Left = 326
Top = 1
Width = 66
Height = 45
Hint = 'Close the application'
Caption = 'Cl&ose'
Glyph.Data = {
Layout = blGlyphTop
NumGlyphs = 2

TextHeight = 13
-----< object \\ MainPanel: TPanel
Left = 0
Top = 0
Width = 401
Height = 49
Align = alTop
ParentShowHint = False
ShowHint = True
TabOrder = 0
-----< object \\ OrderBtn: TSpeedButton
Left = 1
Top = 1
Width = 66
Height = 45
Hint = 'Enter new order'
Caption = '&New Order'
Glyph.Data = {
33333333333333000000}
Layout = blGlyphTop
Spacing = 0
OnClick = NewOrder
end
-----< object \\ BrowseBtn: TSpeedButton
Left = 66
Top = 1
Width = 66
Height = 45
Hint = 'Browse and edit orders'
Caption = '&Browse'
Glyph.Data = {
333333333333333365F0}
Layout = blGlyphTop
Spacing = 0
OnClick = BrowseCustOrd
end
-----< object \\ PartsBtn: TSpeedButton
Left = 131
Top = 1
Width = 66
Height = 45
Hint = 'Browse and edit inventory'

```
      OnClick = CloseApp
    end
  end
  -----< object \\ ViewMenu:
TMenuItem
    Caption = '&View'
    OnClick = ViewMenuClick
    -----< object \\ ViewOrders:
TMenuItem
      Caption = '&Orders...'
      OnClick = BrowseCustOrd
    end
    -----< object \\
ViewPartsInventory: TMenuItem
      Caption = '&Parts/Inventory...'
      OnClick = BrowseParts
    end
    -----< object \\ N7: TMenuItem
      Caption = '-'
    end
    -----< object \\ ViewStayOnTop:
TMenuItem
      Caption = '&Stay On Top'
      OnClick = ToggleStayonTop
    end
    -----< object \\ N1: TMenuItem
      Caption = '-'
    end
    -----< object \\ ViewLocal:
TMenuItem
      Caption = '&Local Data
(Paradox Data)'
      Checked = True
      GroupIndex = 1
      RadioItem = True
      OnClick = ViewLocalClick
    end
    -----< object \\ ViewRemote:
TMenuItem
      Caption = '&Remote Data
(Local Interbase) '
      GroupIndex = 1
      RadioItem = True
      OnClick = ViewRemoteClick
    end
  end
  -----< object \\ HelpMenu:
TMenuItem
    Caption = '&Help'
```

```
      Spacing = 0
      OnClick = CloseApp
    end
  end
  -----< object \\ MainMenu:
TMainMenu
    Left = 36
    Top = 52
    -----< object \\ FileMenu:
TMenuItem
      Caption = '&File'
      -----< object \\ FileNewOrder:
TMenuItem
        Caption = '&New Order...'
        OnClick = NewOrder
      end
      -----< object \\ N5: TMenuItem
        Caption = '-'
      end
      -----< object \\ FilePrintReport:
TMenuItem
        Caption = 'Print &Report'
        -----< object \\ PrintCustList:
TMenuItem
          Caption = '&Customer List'
          OnClick = CustomerReport
        end
        -----< object \\ PrintOrders:
TMenuItem
          Caption = '&Order History...'
          OnClick = OrderReport
        end
        -----< object \\ PrintInvoice:
TMenuItem
          Caption = '&Invoice'
          OnClick = InvoiceReport
        end
      end
      -----< object \\ FilePrinterSetup:
TMenuItem
        Caption = 'Printer Setup...'
        OnClick = PrinterSetupClick
      end
      -----< object \\ N4: TMenuItem
        Caption = '-'
      end
      -----< object \\ FileExit:
TMenuItem
        Caption = 'E&xit'
```

PixelsPerInch = 96
TextHeight = 13
-----< object \\ Panel1: TPanel
Left = 0
Top = 0
Width = 313
Height = 36
Align = alTop
BevelOuter = bvNone
TabOrder = 0
-----< object \\ PrintBtn: TSpeedButton
Left = 270
Top = 5
Width = 25
Height = 25
Hint = 'Print form image'
Glyph.Data = {
OnClick = PrintBtnClick
end
-----< object \\ Bevel1: TBevel
Left = 0
Top = 34
Width = 313
Height = 2
Align = alBottom
Shape = bsTopLine
end
-----< object \\ Navigator: TDBNavigator
Left = 16
Top = 5
Width = 240
Height = 25
HelpContext = 6
DataSource = PartsSource1
ParentShowHint = False
ShowHint = True
TabOrder = 0
end
end
-----< object \\ Panel2: TPanel
Left = 0
Top = 36
Width = 313
Height = 193
Align = alTop
BevelOuter = bvNone

-----< object \\ HelpContents: TMenuItem
Caption = '&Contents'
OnClick = HelpBtnClick
end
-----< object \\ N6: TMenuItem
Caption = '-'
end
-----< object \\ HelpAbout: TMenuItem
Caption = '&About...'
OnClick = AboutClick
end
end
end
-----< object \\ PrinterSetup: TPrinterSetupDialog
Left = 91
Top = 52
end
End
\\
-----< object \\ EdPartsForm: TEdPartsForm
Left = 274
Top = 90
HelpContext = 6
ActiveControl = DBEdit2
BorderIcons = [biSystemMenu, biMinimize]
BorderStyle = bsSingle
Caption = 'Edit Parts'
ClientHeight = 264
ClientWidth = 313
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
OnCloseQuery = FormCloseQuery

```
    Caption = 'ListPrice'
  end
  -----< object \\ Label6: TLabel
    Left = 8
    Top = 122
    Width = 61
    Height = 13
    Caption = 'Backordered'
  end
  -----< object \\ DBEdit2: TDBEdit
    Left = 80
    Top = 29
    Width = 225
    Height = 21
    HelpContext = 6
    DataField = 'Description'
    DataSource = PartsSource1
    TabOrder = 1
  end
  -----< object \\ DBEdit4: TDBEdit
    Left = 80
    Top = 74
    Width = 82
    Height = 21
    HelpContext = 6
    DataField = 'OnHand'
    DataSource = PartsSource1
    TabOrder = 3
  end
  -----< object \\ DBEdit5: TDBEdit
    Left = 80
    Top = 97
    Width = 82
    Height = 21
    HelpContext = 6
    DataField = 'OnOrder'
    DataSource = PartsSource1
    TabOrder = 4
  end
  -----< object \\ DBEdit7: TDBEdit
    Left = 80
    Top = 141
    Width = 102
    Height = 21
    HelpContext = 6
    DataField = 'Cost'
    DataSource = PartsSource1
    TabOrder = 6
```

```
  TabOrder = 1
  -----< object \\ Label1: TLabel
    Left = 8
    Top = 11
    Width = 33
    Height = 13
    Caption = 'PartNo'
  end
  -----< object \\ Label2: TLabel
    Left = 8
    Top = 33
    Width = 53
    Height = 13
    Caption = 'Description'
  end
  -----< object \\ Label3: TLabel
    Left = 8
    Top = 55
    Width = 34
    Height = 13
    Caption = 'Vendor'
  end
  -----< object \\ Label4: TLabel
    Left = 8
    Top = 77
    Width = 40
    Height = 13
    Caption = 'OnHand'
  end
  -----< object \\ Label5: TLabel
    Left = 8
    Top = 100
    Width = 40
    Height = 13
    Caption = 'OnOrder'
  end
  -----< object \\ Label7: TLabel
    Left = 8
    Top = 144
    Width = 21
    Height = 13
    Caption = 'Cost'
  end
  -----< object \\ Label8: TLabel
    Left = 8
    Top = 166
    Width = 40
    Height = 13
```

Left = 147
Top = 235
Width = 75
Height = 25
Caption = 'OK'
Default = True
ModalResult = 1
TabOrder = 2
end
-----< object \\ CancelButton: TButton
Left = 232
Top = 235
Width = 75
Height = 25
Cancel = True
Caption = 'Cancel'
ModalResult = 2
TabOrder = 3
end
-----< object \\ PartsSource1: TDataSource
DataSet = MastData.Parts
Left = 24
Top = 232
end
End
\\
-----< object \\ EdCustForm: TEdCustForm
Left = 253
Top = 134
HelpContext = 1
ActiveControl = DBEdName
BorderIcons = [biSystemMenu, biMinimize]
BorderStyle = bsDialog
Caption = 'Edit Customers'
ClientHeight = 307
ClientWidth = 376
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'

end
-----< object \\ DBEdit8: TDBEdit
Left = 80
Top = 163
Width = 102
Height = 21
HelpContext = 6
DataField = 'ListPrice'
DataSource = PartsSource1
TabOrder = 7
end
-----< object \\ DBEdPartNo: TDBEdit
Left = 80
Top = 6
Width = 102
Height = 21
HelpContext = 6
DataField = 'PartNo'
DataSource = PartsSource1
TabOrder = 0
end
-----< object \\ DBEdit3: TDBEdit
Left = 80
Top = 119
Width = 45
Height = 21
HelpContext = 6
DataField = 'BackOrd'
DataSource = PartsSource1
TabOrder = 5
end
-----< object \\ DataComboBox1: TDBLookupComboBox
Left = 80
Top = 51
Width = 225
Height = 21
DataField = 'VendorNo'
DataSource = PartsSource1
KeyField = 'VendorNo'
ListField = 'VendorName'
ListSource = MastData.VendorSource
TabOrder = 2
end
end
-----< object \\ OKButton: TButton

```
end
-----< object \\ Label6: TLabel
  Left = 124
  Top = 134
  Width = 25
  Height = 13
  Caption = 'State'
end
-----< object \\ Label7: TLabel
  Left = 192
  Top = 134
  Width = 43
  Height = 13
  Caption = 'Zip Code'
end
-----< object \\ Label8: TLabel
  Left = 289
  Top = 132
  Width = 36
  Height = 13
  Caption = 'Country'
end
-----< object \\ Label9: TLabel
  Left = 248
  Top = 53
  Width = 31
  Height = 13
  Caption = 'Phone'
end
-----< object \\ Label11: TLabel
  Left = 284
  Top = 186
  Width = 44
  Height = 13
  Caption = 'Tax Rate'
end
-----< object \\ Label13: TLabel
  Left = 144
  Top = 188
  Width = 58
  Height = 13
  Caption = 'Last Invoice'
end
-----< object \\ Label14: TLabel
  Left = 248
  Top = 92
  Width = 17
  Height = 13
```

```
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
OnCloseQuery = FormCloseQuery
PixelsPerInch = 96
TextHeight = 13
-----< object \\ Panel2: TPanel
  Left = 0
  Top = 39
  Width = 376
  Height = 228
  Align = alClient
  BevelOuter = bvNone
  TabOrder = 0
  -----< object \\ Label1: TLabel
    Left = 248
    Top = 12
    Width = 35
    Height = 13
    Caption = 'CustNo'
  end
  -----< object \\ Label2: TLabel
    Left = 20
    Top = 12
    Width = 44
    Height = 13
    Caption = 'Company'
  end
  -----< object \\ Label3: TLabel
    Left = 20
    Top = 53
    Width = 25
    Height = 13
    Caption = 'Add1'
  end
  -----< object \\ Label4: TLabel
    Left = 20
    Top = 93
    Width = 25
    Height = 13
    Caption = 'Add2'
  end
  -----< object \\ Label5: TLabel
    Left = 20
    Top = 133
    Width = 17
    Height = 13
    Caption = 'City'
```

MastData.CustSource
TabOrder = 1
end
-----< object \\ DBEdit4: TDBEdit
Left = 19
Top = 108
Width = 205
Height = 21
DataField = 'Addr2'
DataSource = MastData.CustSource
TabOrder = 2
end
-----< object \\ DBEdit5: TDBEdit
Left = 19
Top = 148
Width = 98
Height = 21
DataField = 'City'
DataSource = MastData.CustSource
TabOrder = 3
end
-----< object \\ DBEdit6: TDBEdit
Left = 123
Top = 148
Width = 62
Height = 21
DataField = 'State'
DataSource = MastData.CustSource
TabOrder = 4
end
-----< object \\ DBEdit7: TDBEdit
Left = 191
Top = 148
Width = 90
Height = 21
DataField = 'Zip'
DataSource = MastData.CustSource
TabOrder = 5
end
-----< object \\ DBEdit8: TDBEdit
Left = 288
Top = 148
Width = 66
Height = 21

Caption = 'Fax'
end
-----< object \\ Bevel2: TBevel
Left = 8
Top = 178
Width = 345
Height = 5
Shape = bsTopLine
end
-----< object \\ Label10: TLabel
Left = 16
Top = 188
Width = 37
Height = 13
Caption = 'Contact'
end
-----< object \\ DBEdCustNo: TDBEdit
Left = 247
Top = 28
Width = 69
Height = 21
Color = clSilver
DataField = 'CustNo'
DataSource = MastData.CustSource
Enabled = False
ReadOnly = True
TabOrder = 12
end
-----< object \\ DBEdName: TDBEdit
Left = 19
Top = 28
Width = 205
Height = 21
DataField = 'Company'
DataSource = MastData.CustSource
TabOrder = 0
end
-----< object \\ DBEdit3: TDBEdit
Left = 19
Top = 68
Width = 205
Height = 21
DataField = 'Addr1'
DataSource =

```
    Left = 247
    Top = 106
    Width = 106
    Height = 21
    DataField = 'FAX'
    DataSource =
MastData.CustSource
    TabOrder = 8
  end
 end
 -----< object \\ Panel: TPanel
  Left = 0
  Top = 0
  Width = 376
  Height = 39
  Align = alTop
  BevelOuter = bvNone
  BorderWidth = 2
  TabOrder = 1
  -----< object \\ PrintBtn:
TSpeedButton
    Left = 339
    Top = 5
    Width = 25
    Height = 25
    Hint = 'Print form image'
    Glyph.Data = {
      3333333333333300000}
    OnClick = PrintBtnClick
  end
  -----< object \\ Bevel1: TBevel
    Left = 2
    Top = 35
    Width = 372
    Height = 2
    Align = alBottom
    Shape = bsTopLine
  end
  -----< object \\ DBNavigator:
TDBNavigator
    Left = 19
    Top = 5
    Width = 250
    Height = 25
    DataSource =
MastData.CustSource
    ParentShowHint = False
    ShowHint = True
    DataField = 'Country'
    DataSource =
MastData.CustSource
    TabOrder = 6
  end
  -----< object \\ DBEdit9: TDBEdit
    Left = 247
    Top = 68
    Width = 106
    Height = 21
    DataField = 'Phone'
    DataSource =
MastData.CustSource
    TabOrder = 7
  end
  -----< object \\ DBEdit11:
TDBEdit
    Left = 283
    Top = 203
    Width = 54
    Height = 21
    DataField = 'TaxRate'
    DataSource =
MastData.CustSource
    TabOrder = 11
  end
  -----< object \\ DBEdit12:
TDBEdit
    Left = 15
    Top = 203
    Width = 118
    Height = 21
    DataField = 'Contact'
    DataSource =
MastData.CustSource
    TabOrder = 9
  end
  -----< object \\ DBEdInv: TDBEdit
    Left = 143
    Top = 203
    Width = 131
    Height = 21
    DataField = 'LastInvoiceDate'
    DataSource =
MastData.CustSource
    TabOrder = 10
  end
  -----< object \\ DBEdit14:
TDBEdit
```

Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
OnShow = FormShow
PixelsPerInch = 96
TextHeight = 13
-----< object \\ Panel1: TPanel
Left = 0
Top = 0
Width = 384
Height = 37
HelpContext = 4
Align = alTop
BevelOuter = bvNone
BorderWidth = 3
TabOrder = 0
-----< object \\ ActivateBtn: TSpeedButton
Left = 284
Top = 4
Width = 85
Height = 25
AllowAllUp = True
GroupIndex = 1
Caption = 'Backorders'
OnClick = ActivateQuery
end
-----< object \\ Bevel1: TBevel
Left = 3
Top = 32
Width = 378
Height = 2
Align = alBottom
Shape = bsTopLine
end
-----< object \\ Navigator: TDBNavigator
Left = 8
Top = 4
Width = 135
Height = 25
DataSource = MastData.PartsSource

TabOrder = 0
end
end
-----< object \\ Panel1: TPanel
Left = 0
Top = 267
Width = 376
Height = 40
Align = alBottom
BevelOuter = bvNone
TabOrder = 2
-----< object \\ CancelButton: TButton
Left = 292
Top = 8
Width = 75
Height = 25
Cancel = True
Caption = 'Cancel'
ModalResult = 2
TabOrder = 0
end
-----< object \\ OKButton: TButton
Left = 204
Top = 8
Width = 75
Height = 25
Caption = 'OK'
Default = True
ModalResult = 1
TabOrder = 1
end
end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
-----< object \\ BrPartsForm: TBrPartsForm
Left = 229
Top = 151
HelpContext = 4
BorderIcons = [biSystemMenu]
BorderStyle = bsDialog
Caption = 'Browse Parts'
ClientHeight = 235
ClientWidth = 384
Color = clBtnFace

TabOrder = 2
TitleFont.Charset = DEFAULT_CHARSET
TitleFont.Color = clBlack
TitleFont.Height = -11
TitleFont.Name = 'MS Sans Serif'
TitleFont.Style = []
OnDblClick = EditBtnClick
Columns = <
item
Expanded = False
FieldName = 'PartNo'
Visible = True
end
item
Expanded = False
FieldName = 'Description'
Visible = True
end
item
Expanded = False
FieldName = 'OnHand'
Visible = True
end
item
Expanded = False
FieldName = 'OnOrder'
Visible = True
end
item
Expanded = False
FieldName = 'BackOrd'
Visible = True
end>
end
End
\\
-----< object \\ BrDateForm: TBrDateForm
Left = 221
Top = 143
BorderStyle = bsDialog
Caption = 'Select a Date'
ClientHeight = 220
ClientWidth = 362

VisibleButtons = [nbFirst, nbPrior, nbNext, nbLast, nbRefresh]
ParentShowHint = False
ShowHint = True
TabOrder = 0
end
-----< object \\ EditBtn: TButton
Left = 177
Top = 4
Width = 74
Height = 25
Caption = '&Edit'
ModalResult = 1
TabOrder = 1
OnClick = EditBtnClick
end
end
-----< object \\ Panel3: TPanel
Left = 0
Top = 196
Width = 384
Height = 39
Align = alBottom
BevelOuter = bvNone
TabOrder = 1
-----< object \\ CloseBtn: TButton
Left = 300
Top = 9
Width = 74
Height = 25
Cancel = True
Caption = '&Close'
Default = True
ModalResult = 2
TabOrder = 0
OnClick = CloseBtnClick
end
end
-----< object \\ PartsGrid: TDBGrid
Left = 0
Top = 37
Width = 384
Height = 159
Align = alClient
DataSource = MastData.PartsSource
Options = [dgTitles, dgColLines, dgRowLines, dgRowSelect]

```
  Height = 20
  Glyph.Data = {

    33003333333333333300}
  OnClick = NextMonthBtnClick
 end
 -----< object \\ Calendar1:
TCalendar
  Left = 22
  Top = 41
  Width = 318
  Height = 136
  Color = clInfoBk
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = [fsBold]
  ParentFont = False
  StartOfWeek = 0
  TabOrder = 0
  UseCurrentDate = False
  OnChange = Calendar1Change
 end
 -----< object \\ OkBtn: TButton
  Left = 204
  Top = 189
  Width = 73
  Height = 25
  Caption = '&OK'
  Default = True
  ModalResult = 1
  TabOrder = 1
 end
 -----< object \\ CancelBtn: TButton
  Left = 284
  Top = 189
  Width = 73
  Height = 25
  Caption = '&Cancel'
  ModalResult = 2
  TabOrder = 2
 end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\\\\\
```

```
 Color = clBtnFace
 Font.Charset =
DEFAULT_CHARSET
 Font.Color = clBlack
 Font.Height = -11
 Font.Name = 'MS Sans Serif'
 Font.Style = []
 OldCreateOrder = True
 Position = poScreenCenter
 PixelsPerInch = 96
 TextHeight = 13
 -----< object \\ Bevel1: TBevel
  Left = 16
  Top = 8
  Width = 330
  Height = 177
  Shape = bsFrame
 end
 -----< object \\ TitleLabel: TLabel
  Left = 54
  Top = 18
  Width = 257
  Height = 13
  Alignment = taCenter
  AutoSize = False
  Caption = 'February, 1995'
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = [fsBold]
  ParentFont = False
 end
 -----< object \\ PrevMonthBtn:
TSpeedButton
  Left = 22
  Top = 16
  Width = 20
  Height = 20
  Glyph.Data = {
  OnClick = PrevMonthBtnClick
 end
 -----< object \\ NextMonthBtn:
TSpeedButton
  Left = 320
  Top = 16
  Width = 20
```

Font.Style = [fsBold]
ParentFont = False
IsControl = True
end
-----< object \\ Version: TLabel
Left = 128
Top = 40
Width = 65
Height = 13
Caption = 'Version 3.0'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
IsControl = True
end
-----< object \\ Copyright: TLabel
Left = 16
Top = 136
Width = 258
Height = 13
Caption = 'Copyright 1995,1997 Marine Adventures, Inc.'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentFont = False
IsControl = True
end
end
-----< object \\ Button1: TButton
Left = 114
Top = 180
Width = 75
Height = 25
Cancel = True
Caption = '&OK'
Default = True
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11

-----< object \\ AboutBox: TAboutBox
Left = 239
Top = 127
BorderStyle = bsDialog
Caption = 'About'
ClientHeight = 210
ClientWidth = 304
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clWindowText
Font.Height = -13
Font.Name = 'System'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
PixelsPerInch = 96
TextHeight = 16
-----< object \\ Panel1: TPanel
Left = 8
Top = 8
Width = 289
Height = 161
BevelInner = bvRaised
BevelOuter = bvLowered
TabOrder = 0
-----< object \\ ProgramIcon: TImage
Left = 8
Top = 8
Width = 105
Height = 105
Picture.Data = {
IsControl = True
end
-----< object \\ ProductName: TLabel
Left = 128
Top = 16
Width = 35
Height = 13
Caption = 'MAST'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'

```
  end
  -----< object \\ OKBtn: TButton
    Left = 2
    Top = 160
    Width = 75
    Height = 25
    Caption = '&OK'
    Default = True
    ModalResult = 1
    TabOrder = 1
  end
  -----< object \\ DBLookupListBox1: TDBLookupListBox
    Left = 5
    Top = 16
    Width = 153
    Height = 134
    KeyField = 'OrderNo'
    ListField = 'OrderNo'
    ListSource = MastData.OrdersSource
    TabOrder = 2
  end
End

انتهى الملف النصي  ع ل
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

```

```
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ModalResult = 1
    ParentFont = False
    TabOrder = 1
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

-----< object \\ PickOrderNoDlg: TPickOrderNoDlg
  Left = 236
  Top = 124
  AutoScroll = False
  Caption = 'Order No.'
  ClientHeight = 190
  ClientWidth = 166
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  OnShow = FormShow
  PixelsPerInch = 96
  TextHeight = 13
  -----< object \\ Label1: TLabel
    Left = 5
    Top = 1
    Width = 94
    Height = 13
    Caption = '&Select an Order No.'
    FocusControl = DBLookupListBox1
  end
  -----< object \\ CancelBtn: TButton
    Left = 86
    Top = 160
    Width = 75
    Height = 25
    Cancel = True
    Caption = '&Cancel'
    ModalResult = 2
    TabOrder = 0
```

```
unit SrchDlg;

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, DBTables, DB, StdCtrls, ExtCtrls, Grids, DBGrids, Buttons;

type
  TSearchDlg = class(TForm)
    DataSource: TDataSource;
    DBGrid1: TDBGrid;
    OKBtn: TButton;
    CancelBtn: TButton;
    SearchEd: TEdit;
    OrderCombo: TComboBox;
    Label1: TLabel;
    Label2: TLabel;
    SearchButton: TSpeedButton;
<------>    procedure DBGrid1DblClick(Sender: TObject);
<------>    procedure SearchButtonClick(Sender: TObject);
<------>    procedure OrderComboChange(Sender: TObject);
<------>    procedure SearchEdKeyPress(Sender: TObject; var Key: Char);
<------>    procedure SearchEdChange(Sender: TObject);
<------>    procedure FormCreate(Sender: TObject);
  private
    SrchFld: TField;
    function GetCustNo: Double;
<------>    procedure SetCustNo(NewCustNo: Double);
    function GetPartNo: Double;
<------>    procedure SetPartNo(NewPartNo: Double);
  public
```

Procedures 43

الملف التنفيذي لبرنامج

(MastApp)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

```
unit Splash;

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, ExtCtrls;

type
  TSplashForm = class(TForm)
    Panel1: TPanel;
    Label3: TLabel;
    Bevel1: TBevel;
    Label1: TLabel;
    Image1: TImage;
<------>    procedure Panel1Click(Sender: TObject);
  end;

var
  SplashForm: TSplashForm;

implementation

{$R *.dfm}

procedure TSplashForm.Panel1Click(Sender: TObject);
begin

end;

end.
```

function TSearchDlg.ShowModalCust: Integer;
begin
OrderCombo.Items.Clear;
OrderCombo.Items.Add('Company');
OrderCombo.Items.Add('CustNo');
OrderCombo.ItemIndex := 0;
Datasource.Dataset := MastData.Cust;
OrderComboChange(nil);
Caption := 'Select a Customer';
Result := ShowModal;
end;
function TSearchDlg.ShowModalParts: Integer;
begin
OrderCombo.Items.Clear;
OrderCombo.Items.Add('Description');
OrderCombo.Items.Add('PartNo');
OrderCombo.ItemIndex := 0;
Datasource.Dataset := MastData.Parts;
OrderComboChange(nil);
Caption := 'Select a Part';
Result := ShowModal;
end;
procedure TSearchDlg.DBGrid1DblClick(Sender: TObject);
begin
ModalResult := mrOK;
end;
procedure TSearchDlg.SearchButtonClick(Sender: TObject);
begin
if not Datasource.Dataset.Locate(OrderCombo.Text, SearchEd.Text,
[loCaseInsensitive, loPartialKey])

property PartNo: Double read GetPartNo write SetPartNo;
property CustNo: Double read GetCustNo write SetCustNo;
function ShowModalCust: Integer;
function ShowModalParts: Integer;
end;
var
SearchDlg: TSearchDlg;
implementation
uses DataMod;
{$R *.dfm}
function TSearchDlg.GetCustNo: Double;
begin
Result := MastData.CustCustNo.Value;
end;
procedure TSearchDlg.SetCustNo(NewCustNo: Double);
begin
MastData.Cust.Locate('CustNo', NewCustNo, []);
end;
function TSearchDlg.GetPartNo: Double;
begin
Result := MastData.PartsPartNo.Value;
end;
procedure TSearchDlg.SetPartNo(NewPartNo: Double);
begin
MastData.Parts.Locate('PartNo', NewPartNo, []);
end;

```

interface

uses
  SysUtils, Windows, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, DBTables, DB, StdCtrls, ExtCtrls, Grids, DBGrids, Buttons;

type
  TSearchDlg = class(TForm)
    DataSource: TDataSource;
    DBGrid1: TDBGrid;
    OKBtn: TButton;
    CancelBtn: TButton;
    SearchEd: TEdit;
    OrderCombo: TComboBox;
    Label1: TLabel;
    Label2: TLabel;
    SearchButton: TSpeedButton;
<------>    procedure DBGrid1DblClick(Sender: TObject);
<------>    procedure SearchButtonClick(Sender: TObject);
<------>    procedure OrderComboChange(Sender: TObject);
<------>    procedure SearchEdKeyPress(Sender: TObject; var Key: Char);
<------>    procedure SearchEdChange(Sender: TObject);
<------>    procedure FormCreate(Sender: TObject);
  private
    SrchFld: TField;
    function GetCustNo: Double;
<------>    procedure SetCustNo(NewCustNo: Double);
    function GetPartNo: Double;
<------>    procedure SetPartNo(NewPartNo: Double);
  public
    property PartNo: Double read GetPartNo write SetPartNo;
    property CustNo: Double read
```

```
then
    MessageDlg('No matching record found.', mtInformation, [mbOK], 0);
end;

procedure TSearchDlg.OrderComboChange(Sender: TObject);
begin
  SrchFld := Datasource.Dataset.FieldByName(OrderCombo.Text);
  SearchEd.Text := '';
end;

procedure TSearchDlg.SearchEdKeyPress(Sender: TObject; var Key: Char);
begin
  if Assigned(SrchFld) and (Key > ' ') and not (SrchFld.IsValidChar(Key)) then
  begin
    MessageBeep(0);
    Key := #0;
  end;
end;

procedure TSearchDlg.SearchEdChange(Sender: TObject);
begin
  SearchButton.Enabled := SearchEd.Text <> '';
end;

procedure TSearchDlg.FormCreate(Sender: TObject);
begin

end;

end.

.


unit SrchDlg;
```

```
begin
  OrderCombo.Items.Clear;
  OrderCombo.Items.Add('Company');
  OrderCombo.Items.Add('CustNo');
  OrderCombo.ItemIndex := 0;
  Datasource.Dataset := MastData.Cust;
  OrderComboChange(nil);
  Caption := 'Select a Customer';
  Result := ShowModal;
end;

function TSearchDlg.ShowModalParts: Integer;
begin
  OrderCombo.Items.Clear;
  OrderCombo.Items.Add('Description');
  OrderCombo.Items.Add('PartNo');
  OrderCombo.ItemIndex := 0;
  Datasource.Dataset := MastData.Parts;
  OrderComboChange(nil);
  Caption := 'Select a Part';
  Result := ShowModal;
end;

procedure TSearchDlg.DBGrid1DblClick(Sender: TObject);
begin
  ModalResult := mrOK;
end;

procedure TSearchDlg.SearchButtonClick(Sender: TObject);
begin
  if not Datasource.Dataset.Locate(OrderCombo.Text, SearchEd.Text,
    [loCaseInsensitive, loPartialKey]) then
      MessageDlg('No matching record found.', mtInformation, [mbOK], 0);
```

```
GetCustNo write SetCustNo;
    function ShowModalCust: Integer;
    function ShowModalParts: Integer;
  end;

var
  SearchDlg: TSearchDlg;

implementation

uses DataMod;

{$R *.dfm}

function TSearchDlg.GetCustNo: Double;
begin
  Result := MastData.CustCustNo.Value;
end;

procedure TSearchDlg.SetCustNo(NewCustNo: Double);
begin
  MastData.Cust.Locate('CustNo', NewCustNo, []);
end;

function TSearchDlg.GetPartNo: Double;
begin
  Result := MastData.PartsPartNo.Value;
end;

procedure TSearchDlg.SetPartNo(NewPartNo: Double);
begin
  MastData.Parts.Locate('PartNo', NewPartNo, []);
end;

function TSearchDlg.ShowModalCust: Integer;
```

```
Classes, Graphics, Controls,
  Forms, Dialogs, DBTables, DB, StdCtrls, ExtCtrls, Grids, DBGrids, Buttons;

type
  TSearchDlg = class(TForm)
    DataSource: TDataSource;
    DBGrid1: TDBGrid;
    OKBtn: TButton;
    CancelBtn: TButton;
    SearchEd: TEdit;
    OrderCombo: TComboBox;
    Label1: TLabel;
    Label2: TLabel;
    SearchButton: TSpeedButton;
<------>    procedure DBGrid1DblClick(Sender: TObject);
<------>    procedure SearchButtonClick(Sender: TObject);
<------>    procedure OrderComboChange(Sender: TObject);
<------>    procedure SearchEdKeyPress(Sender: TObject; var Key: Char);
<------>    procedure SearchEdChange(Sender: TObject);
<------>    procedure FormCreate(Sender: TObject);
  private
    SrchFld: TField;
    function GetCustNo: Double;
<------>    procedure SetCustNo(NewCustNo: Double);
    function GetPartNo: Double;
<------>    procedure SetPartNo(NewPartNo: Double);
  public
    property PartNo: Double read GetPartNo write SetPartNo;
    property CustNo: Double read GetCustNo write SetCustNo;
    function ShowModalCust: Integer;
    function ShowModalParts: Integer;
```

```
end;

procedure TSearchDlg.OrderComboChange(Sender: TObject);
begin
  SrchFld := Datasource.Dataset.FieldByName(OrderCombo.Text);
  SearchEd.Text := '';
end;

procedure TSearchDlg.SearchEdKeyPress(Sender: TObject; var Key: Char);
begin
  if Assigned(SrchFld) and (Key > ' ') and not (SrchFld.IsValidChar(Key)) then
  begin
    MessageBeep(0);
    Key := #0;
  end;
end;

procedure TSearchDlg.SearchEdChange(Sender: TObject);
begin
  SearchButton.Enabled := SearchEd.Text <> '';
end;

procedure TSearchDlg.FormCreate(Sender: TObject);
begin

end;

end.


unit SrchDlg;

interface

uses
  SysUtils, Windows, Messages,
```

```
  OrderCombo.Items.Add('CustNo');
  OrderCombo.ItemIndex := 0;
  Datasource.Dataset :=
MastData.Cust;
  OrderComboChange(nil);
  Caption := 'Select a Customer';
  Result := ShowModal;
end;

function
TSearchDlg.ShowModalParts:
Integer;
begin
  OrderCombo.Items.Clear;

OrderCombo.Items.Add('Descriptio
n');
  OrderCombo.Items.Add('PartNo');
  OrderCombo.ItemIndex := 0;
  Datasource.Dataset :=
MastData.Parts;
  OrderComboChange(nil);
  Caption := 'Select a Part';
  Result := ShowModal;
end;

procedure
TSearchDlg.DBGrid1DblClick(Send
er: TObject);
begin
  ModalResult := mrOK;
end;

procedure
TSearchDlg.SearchButtonClick(Sen
der: TObject);
begin
  if not
Datasource.Dataset.Locate(OrderCo
mbo.Text, SearchEd.Text,
    [loCaseInsensitive, loPartialKey])
then
     MessageDlg('No matching record
found.', mtInformation, [mbOK], 0);
end;

procedure
TSearchDlg.OrderComboChange(Se
nder: TObject);
```

```
  end;

var
  SearchDlg: TSearchDlg;

implementation

uses DataMod;

{$R *.dfm}

function TSearchDlg.GetCustNo:
Double;
begin
  Result :=
MastData.CustCustNo.Value;
end;

procedure
TSearchDlg.SetCustNo(NewCustNo:
Double);
begin
  MastData.Cust.Locate('CustNo',
NewCustNo, []);
end;

function TSearchDlg.GetPartNo:
Double;
begin
  Result :=
MastData.PartsPartNo.Value;
end;

procedure
TSearchDlg.SetPartNo(NewPartNo:
Double);
begin
  MastData.Parts.Locate('PartNo',
NewPartNo, []);
end;

function
TSearchDlg.ShowModalCust:
Integer;
begin
  OrderCombo.Items.Clear;

OrderCombo.Items.Add('Company'
);
```

```
type
  TQueryCustDlg = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    FromEdit: TEdit;
    ToEdit: TEdit;
    CancelBtn: TButton;
    OkBtn: TButton;
    Msglab: TLabel;
    PopupCalBtnFrom:
TSpeedButton;
    PopupCalToBtn: TSpeedButton;
    Bevel1: TBevel;
<------>    procedure
OkBtnClick(Sender: TObject);
<------>    procedure
PopupCalBtnFromClick(Sender:
TObject);
<------>    procedure
PopupCalToBtnClick(Sender:
TObject);
<------>    procedure
FormCreate(Sender: TObject);
  private
    function GetFromDate:
TDateTime;
    function GetToDate: TDateTime;
<------>    procedure
SetFromDate(NewDate:
TDateTime);
<------>    procedure
SetToDate(NewDate: TDateTime);
  public
    property FromDate: TDateTime
read GetFromDate write
SetFromDate;
    property ToDate: TDateTime
read GetToDate write SetToDate;
  end;

var
  QueryCustDlg: TQueryCustDlg;

implementation

{$R *.dfm}

uses Pickdate;

```

```
begin
  SrchFld :=
Datasource.Dataset.FieldByName(O
rderCombo.Text);
  SearchEd.Text := '';
end;

procedure
TSearchDlg.SearchEdKeyPress(Sen
der: TObject; var Key: Char);
begin
  if Assigned(SrchFld) and (Key > ' ')
and not (SrchFld.IsValidChar(Key))
then
  begin
    MessageBeep(0);
    Key := #0;
  end;
end;

procedure
TSearchDlg.SearchEdChange(Sende
r: TObject);
begin
  SearchButton.Enabled :=
SearchEd.Text <> '';
end;

procedure
TSearchDlg.FormCreate(Sender:
TObject);
begin

end;

end.



unit QryCust;

interface

uses
  SysUtils, Windows, Messages,
Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, ExtCtrls,
Buttons;

```

```
    ShowMessage('  Invalid date specified');
    ModalResult := mrNone;
  end;
end;

procedure TQueryCustDlg.PopupCalBtnFromClick(Sender: TObject);
begin
  BrDateForm.Date := StrToDate(FromEdit.Text);  { start with current date }
  if BrDateForm.ShowModal = mrOk then
    FromEdit.Text := DateToStr(BrDateForm.Date);
end;

procedure TQueryCustDlg.PopupCalToBtnClick(Sender: TObject);
begin
  BrDateForm.Date := StrToDate(ToEdit.Text);    { start with current date }
  if BrDateForm.ShowModal = mrOk then
    ToEdit.Text := DateToStr(BrDateForm.Date);
end;

procedure TQueryCustDlg.FormCreate(Sender: TObject);
begin
  MsgLab.Caption := 'Customers with LastInvoiceDate ranging:';
  FromDate := EncodeDate(1995, 01, 01);
  ToDate := Now;
end;

end.



unit Pickrep;

procedure TQueryCustDlg.SetFromDate(NewDate: TDateTime);
begin
  FromEdit.Text := DateToStr(NewDate);
end;

procedure TQueryCustDlg.SetToDate(NewDate: TDateTime);
begin
  ToEdit.Text := DateToStr(NewDate);
end;

function TQueryCustDlg.GetFromDate: TDateTime;
begin
  if FromEdit.Text = '' then Result := 0
  else Result := StrToDate(FromEdit.Text);
end;

function TQueryCustDlg.GetToDate: TDateTime;
begin
  if ToEdit.Text = '' then Result := 0
  else Result := StrToDate(ToEdit.Text);
end;

procedure TQueryCustDlg.OkBtnClick(Sender: TObject);
begin
  try
    if (ToDate <> 0) and (ToDate < FromDate) then
    begin
      ShowMessage('"TO" date cannot be less than "FROM" date');
      ModalResult := mrNone;
    end
    else ModalResult := mrOk;
  except
```

```
end.


unit Pickrep;

interface

uses Windows, Classes, Graphics, Forms, Controls, Buttons,
  StdCtrls, ExtCtrls;

type
  TPickRpt = class(TForm)
    OKBtn: TButton;
    CloseBtn: TButton;
    ViewBtn: TButton;
    ReportType: TRadioGroup;
<------>    procedure OKBtnClick(Sender: TObject);
<------>    procedure ViewBtnClick(Sender: TObject);
<------>    procedure FormCreate(Sender: TObject);
  public
    Preview : Boolean;
  end;

var
  PickRpt: TPickRpt;

implementation

{$R *.dfm}

procedure TPickRpt.OKBtnClick(Sender: TObject);
begin
  Preview := False;
end;

procedure TPickRpt.ViewBtnClick(Sender: TObject);
begin
  Preview := True;
end;
```

```
interface

uses Windows, Classes, Graphics, Forms, Controls, Buttons,
  StdCtrls, ExtCtrls;

type
  TPickRpt = class(TForm)
    OKBtn: TButton;
    CloseBtn: TButton;
    ViewBtn: TButton;
    ReportType: TRadioGroup;
<------>    procedure OKBtnClick(Sender: TObject);
<------>    procedure ViewBtnClick(Sender: TObject);
<------>    procedure FormCreate(Sender: TObject);
  public
    Preview : Boolean;
  end;

var
  PickRpt: TPickRpt;

implementation

{$R *.dfm}

procedure TPickRpt.OKBtnClick(Sender: TObject);
begin
  Preview := False;
end;

procedure TPickRpt.ViewBtnClick(Sender: TObject);
begin
  Preview := True;
end;

procedure TPickRpt.FormCreate(Sender: TObject);
begin

end;
```

```
procedure TPickRpt.ViewBtnClick(Sender: TObject);
begin
  Preview := True;
end;

procedure TPickRpt.FormCreate(Sender: TObject);
begin

end;

end.


unit PickInvc;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, DBCtrls;

type
  TPickOrderNoDlg = class(TForm)
    CancelBtn: TButton;
    OKBtn: TButton;
    DBLookupListBox1: TDBLookupListBox;
    Label1: TLabel;
<------>    procedure FormShow(Sender: TObject);
<------>    procedure DBLookupListBox1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  PickOrderNoDlg: TPickOrderNoDlg;
```

```
procedure TPickRpt.FormCreate(Sender: TObject);
begin

end;

end.



unit Pickrep;

interface

uses Windows, Classes, Graphics, Forms, Controls, Buttons,
  StdCtrls, ExtCtrls;

type
  TPickRpt = class(TForm)
    OKBtn: TButton;
    CloseBtn: TButton;
    ViewBtn: TButton;
    ReportType: TRadioGroup;
<------>    procedure OKBtnClick(Sender: TObject);
<------>    procedure ViewBtnClick(Sender: TObject);
<------>    procedure FormCreate(Sender: TObject);
  public
    Preview : Boolean;
  end;

var
  PickRpt: TPickRpt;

implementation

{$R *.dfm}

procedure TPickRpt.OKBtnClick(Sender: TObject);
begin
  Preview := False;
end;
```

```
TOrdersByDateReport;

implementation

{$R *.dfm}

procedure TOrdersByDateReport.FormCreate(Sender: TObject);
begin

end;

end.
```

```

implementation

uses DataMod;

{$R *.dfm}

procedure TPickOrderNoDlg.FormShow(Sender: TObject);
begin
  MastData.Orders.Open;
end;

procedure TPickOrderNoDlg.DBLookupListBox1Click(Sender: TObject);
begin

end;

end.





unit OrderRpt;

interface

uses SysUtils, Windows, Messages, Classes, Graphics, Controls,
  StdCtrls, ExtCtrls, Forms, Quickrpt, QRCtrls;

type
  TOrdersByDateReport = class(TQuickRep)
<------>    procedure FormCreate(Sender: TObject);
  private

  public

  end;

var
  OrdersByDateReport:
```

Objects643
الملف لبرنامج
(TeeChart)
إعداد علاء الدين اللباد
هاتف ٠٩٤٤٥٧٥٣٧١

برنامج يتعلق بالرسوم البيانية والإحصائية يحتوي العديد من الميزات الرائعة في برنامج دلفي ٧ وهو موجود كمثال باسم ضمن البرنامج نفسه ويمكن الاستفادة من جميع الميزات الموجودة فيه قام بتحليل وشرح البرنامج علاء الدين اللباد ٠٩٤٤٥٧٥٣٧١
الملف النصي للبرنامج ويحتوي على ٦٤٣ عنصرا
Objects 643
وجميعها معلمة بالسهم
<----->

<-------> object BubbleForm: TBubbleForm
Left = 191
Top = 113
Width = 661
Height = 439
Caption = 'TeeChart Bubble Series Demo'
Color = clBtnFace
ParentFont = True
OldCreateOrder = True
Position = poScreenCenter
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> object Chart1: TChart
Left = 153
Top = 0
Width = 500
Height = 405
AnimatedZoom = True
AnimatedZoomSteps = 3
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear

BackWall.Color = clWhite
Foot.Alignment = taLeftJustify
Foot.Font.Charset = DEFAULT_CHARSET
Foot.Font.Color = clRed
Foot.Font.Height = -12
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsItalic]
Foot.Frame.Color = clScrollBar
Foot.Text.Strings = (
'Left drag (Up/Down) to Zoom. Right drag to Scroll. Invert Zoom r' +
'ectangle to reset.')
Title.Alignment = taRightJustify
Title.Font.Charset = DEFAULT_CHARSET
Title.Font.Color = clBlue
Title.Font.Height = -15
Title.Font.Name = 'Arial'
Title.Font.Style = [fsItalic]
Title.Frame.Color = clScrollBar
Title.Text.Strings = (
'This demo shows the Bubble chart Type.'
'Each Point is a Bubble with a custom Radius, Color and Label.')
BackColor = clWhite
BottomAxis.LabelsAngle = 90
BottomAxis.LabelsFont.Charset = DEFAULT_CHARSET
BottomAxis.LabelsFont.Color = clTeal
BottomAxis.LabelsFont.Height = -12
BottomAxis.LabelsFont.Name = 'Arial'
BottomAxis.LabelsFont.Style = []
BottomAxis.Title.Font.Charset = DEFAULT_CHARSET
BottomAxis.Title.Font.Color = clGreen
BottomAxis.Title.Font.Height = -16
BottomAxis.Title.Font.Name = 'Arial'
BottomAxis.Title.Font.Style = [fsItalic]

```
    Pointer.HorizSize = 57
    Pointer.InflateMargins = False
    Pointer.Style = psCircle
    Pointer.VertSize = 57
    Pointer.Visible = True
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier =
1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier =
1.000000000000000000
    YValues.Order = loNone
    RadiusValues.DateTime = False
    RadiusValues.Name = 'Radius'
    RadiusValues.Multiplier =
1.000000000000000000
    RadiusValues.Order = loNone
  end
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 0
  Width = 153
  Height = 405
  Align = alLeft
  TabOrder = 1
  <-------> object Label1: TLabel
    Left = 6
    Top = 37
    Width = 72
    Height = 17
    HelpType = htKeyword
    Alignment = taRightJustify
    AutoSize = False
    Caption = 'Bubble &Style:'
  end
  <-------> object ZoomInButton:
TSpeedButton
    Left = 40
    Top = 328
    Width = 25
    Height = 25
    HelpType = htKeyword
    Glyph.Data = {
    NumGlyphs = 2
```

```
  LeftAxis.LabelsFont.Charset =
DEFAULT_CHARSET
  LeftAxis.LabelsFont.Color =
clWhite
  LeftAxis.LabelsFont.Height = -13
  LeftAxis.LabelsFont.Name =
'Arial'
  LeftAxis.LabelsFont.Style = []
  LeftAxis.MinorTickLength = 4
  LeftAxis.MinorTicks.Color =
4227327
  LeftAxis.TickLength = 8
  LeftAxis.Title.Angle = 0
  LeftAxis.Title.Caption = 'Stock
Price'
  LeftAxis.Title.Font.Charset =
DEFAULT_CHARSET
  LeftAxis.Title.Font.Color =
clNavy
  LeftAxis.Title.Font.Height = -15
  LeftAxis.Title.Font.Name = 'Arial'
  LeftAxis.Title.Font.Style =
[fsBold]
  Legend.ColorWidth = 32
  Legend.Visible = False
  RightAxis.Title.Angle = 180
  TopAxis.Title.Caption = 'Date'
  View3D = False
  Align = alClient
  BevelWidth = 0
  TabOrder = 0
  <-------> object BubbleSeries1:
TBubbleSeries
    HorizAxis = aTopAxis
    Marks.ArrowLength = 0
    Marks.Clip = True
    Marks.Font.Charset =
DEFAULT_CHARSET
    Marks.Font.Color = clWhite
    Marks.Font.Height = -16
    Marks.Font.Name = 'Arial'
    Marks.Font.Style = [fsItalic]
    Marks.Frame.Color = 8454143
    Marks.Frame.Visible = False
    Marks.Transparent = True
    Marks.Visible = True
    SeriesColor = clRed
    OnGetPointerStyle =
BubbleSeries1GetPointerStyle
```

```
    Style = csDropDownList
    ItemHeight = 13
    TabOrder = 2
    OnChange = ComboBox1Change
    Items.Strings = (
      'Rectangle'
      'Circle'
      'Triangle'
      'DownTriangle'
      'Cross'
      'DiagCross'
      'Star'
      'Diamond')
  end
  <-------> object BitBtn3: TBitBtn
    Left = 28
    Top = 364
    Width = 89
    Height = 33
    TabOrder = 3
    Kind = bkClose
  end
  <-------> object CheckBox3:
TCheckBox
    Left = 16
    Top = 88
    Width = 97
    Height = 17
    Caption = '&Random'
    TabOrder = 4
    OnClick = CheckBox3Click
  end
  <-------> object Memo1: TMemo
    Left = 16
    Top = 136
    Width = 121
    Height = 185
    Lines.Strings = (
      'Each "Bubble" is '
      'represented with a '
      'center position and '
      'Radius.'
      'Each point can have a '
      'different style using the '
      'OnGetPointerStyle '
      'event.'
      ''
      'This example also '
```

```
    OnClick = ZoomInButtonClick
  end
  <-------> object ZoomOutButton:
TSpeedButton
    Left = 72
    Top = 328
    Width = 25
    Height = 25
    HelpType = htKeyword
    Glyph.Data = {
    NumGlyphs = 2
    OnClick = ZoomOutButtonClick
  end
  <-------> object CheckBox1:
TCheckBox
    Left = 17
    Top = 8
    Width = 128
    Height = 17
    Caption = '&Animate !!!'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -19
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 0
    OnClick = CheckBox1Click
  end
  <-------> object CheckBox2:
TCheckBox
    Left = 16
    Top = 104
    Width = 93
    Height = 17
    Caption = 'View &Marks'
    Checked = True
    State = cbChecked
    TabOrder = 1
    OnClick = CheckBox2Click
  end
  <-------> object ComboBox1:
TComboBox
    Left = 14
    Top = 53
    Width = 131
    Height = 21
```

```
   Foot.Font.Name = 'Arial'
   Foot.Font.Style = [fsItalic]
   Foot.Frame.Color = clScrollBar
   Foot.Text.Strings = (
     ''
     '')
   Foot.Visible = False
   LeftWall.Color = 16777088
   LeftWall.Size = 5
   MarginBottom = 7
   MarginLeft = 5
   Title.Font.Charset = DEFAULT_CHARSET
   Title.Font.Color = clGreen
   Title.Font.Height = -16
   Title.Font.Name = 'Arial'
   Title.Font.Style = [fsBold, fsItalic]
   Title.Frame.Color = clScrollBar
   Title.Text.Strings = (
     'Line Series')
   BottomAxis.Axis.Width = 1
   BottomAxis.LabelsFont.Charset = DEFAULT_CHARSET
   BottomAxis.LabelsFont.Color = clBlue
   BottomAxis.LabelsFont.Height = -12
   BottomAxis.LabelsFont.Name = 'Arial'
   BottomAxis.LabelsFont.Style = []
   BottomAxis.Title.Font.Charset = DEFAULT_CHARSET
   BottomAxis.Title.Font.Color = clGreen
   BottomAxis.Title.Font.Height = -16
   BottomAxis.Title.Font.Name = 'Arial'
   BottomAxis.Title.Font.Style = [fsItalic]
   Chart3DPercent = 30
   LeftAxis.Axis.Width = 1
   LeftAxis.LabelsFont.Charset = DEFAULT_CHARSET
   LeftAxis.LabelsFont.Color = clPurple
   LeftAxis.LabelsFont.Height = -13
   LeftAxis.LabelsFont.Name = 'Arial'
        'shows '
        'Axis labels and title '
        'rotation.')
      TabOrder = 5
    end
  end
  <-------> object Timer1: TTimer
    Enabled = False
    Interval = 1
    OnTimer = Timer1Timer
    Left = 170
    Top = 86
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

<-------> object BasicForm: TBasicForm
  Left = 158
  Top = 114
  Width = 746
  Height = 487
  Caption = 'TeeChart Basic Series Demo'
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  WindowState = wsMaximized
  OnCreate = FormCreate
  OnResize = FormResize
  OnShow = FormShow
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 128
    Top = 49
    Width = 329
    Height = 200
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BottomWall.Color = 16777088
    BottomWall.Size = 6
    Foot.Alignment = taLeftJustify
    Foot.Font.Charset = DEFAULT_CHARSET
    Foot.Font.Color = clRed
    Foot.Font.Height = -12
```

```
    Marks.Visible = False
    SeriesColor = clBlue
    LinePen.Color = clBlue
    Pointer.InflateMargins = True
    Pointer.Style = psRectangle
    Pointer.Visible = False
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
  end
 end
 <-------> object Chart2: TChart
  Left = 128
  Top = 256
  Width = 329
  Height = 173
  BackWall.Brush.Color = clWhite
  BackWall.Brush.Style = bsClear
  Foot.Alignment = taLeftJustify
  Foot.Font.Charset = DEFAULT_CHARSET
  Foot.Font.Color = clBlack
  Foot.Font.Height = -12
  Foot.Font.Name = 'Arial'
  Foot.Font.Style = [fsItalic]
  Foot.Frame.Color = clScrollBar
  Foot.Text.Strings = (
    '')
  Foot.Visible = False
  Gradient.Direction = gdRightLeft
  Gradient.EndColor = clWhite
  Gradient.StartColor = clYellow
  LeftWall.Color = clWhite
  MarginLeft = 7
  MarginRight = 7
  MarginTop = 5
  Title.Font.Charset = DEFAULT_CHARSET
  Title.Font.Color = clBlue
  Title.Font.Height = -16
  Title.Font.Name = 'Arial'
  LeftAxis.LabelsFont.Style = []
  LeftAxis.Title.Font.Charset = DEFAULT_CHARSET
  LeftAxis.Title.Font.Color = clNavy
  LeftAxis.Title.Font.Height = -15
  LeftAxis.Title.Font.Name = 'Arial'
  LeftAxis.Title.Font.Style = [fsBold]
  Legend.Alignment = laBottom
  Legend.Color = clBlack
  Legend.ColorWidth = 16
  Legend.Font.Charset = DEFAULT_CHARSET
  Legend.Font.Color = clWhite
  Legend.Font.Height = -12
  Legend.Font.Name = 'Arial'
  Legend.Font.Style = [fsItalic]
  Legend.ShadowColor = clGray
  RightAxis.Visible = False
  TopAxis.Visible = False
  BorderWidth = 1
  Color = 8454016
  TabOrder = 0
  <-------> object Winter: TLineSeries
    Marks.ArrowLength = 8
    Marks.Frame.Color = 8454143
    Marks.Visible = False
    SeriesColor = clRed
    LinePen.Color = clRed
    Pointer.InflateMargins = True
    Pointer.Style = psRectangle
    Pointer.Visible = False
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
  end
  <-------> object Summer: TLineSeries
    Marks.ArrowLength = 8
    Marks.Frame.Color = 8454143
```

TopAxis.LabelsAngle = 90
TopAxis.LabelsFont.Charset = DEFAULT_CHARSET
TopAxis.LabelsFont.Color = clRed
TopAxis.LabelsFont.Height = -11
TopAxis.LabelsFont.Name = 'Arial'
TopAxis.LabelsFont.Style = []
TopAxis.LabelsSize = 30
TabOrder = 3
<-------> object BarSeries1: TBarSeries
Marks.ArrowLength = 20
Marks.BackColor = 16777088
Marks.Font.Charset = DEFAULT_CHARSET
Marks.Font.Color = clBlack
Marks.Font.Height = -8
Marks.Font.Name = 'MS Serif'
Marks.Font.Style = []
Marks.Style = smsPercent
Marks.Visible = True
SeriesColor = clRed
BarPen.Visible = False
MultiBar = mbNone
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 142
Top = 274
end
end
<-------> object Chart3: TChart
Left = 464
Top = 49
Width = 309
Height = 200
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
BackWall.Color = clSilver
Foot.Alignment = taLeftJustify

Title.Font.Style = [fsBold, fsItalic]
Title.Frame.Color = clScrollBar
Title.Text.Strings = (
'Bar Series')
BottomAxis.Axis.Width = 1
BottomAxis.DateTimeFormat = 'MM/yy'
BottomAxis.LabelsFont.Charset = DEFAULT_CHARSET
BottomAxis.LabelsFont.Color = clBlue
BottomAxis.LabelsFont.Height = -11
BottomAxis.LabelsFont.Name = 'Arial'
BottomAxis.LabelsFont.Style = []
BottomAxis.LabelsSeparation = 20
BottomAxis.Title.Font.Charset = DEFAULT_CHARSET
BottomAxis.Title.Font.Color = clBlue
BottomAxis.Title.Font.Height = -16
BottomAxis.Title.Font.Name = 'Arial'
BottomAxis.Title.Font.Style = [fsItalic]
Chart3DPercent = 80
LeftAxis.Axis.Width = 1
LeftAxis.LabelsFont.Charset = DEFAULT_CHARSET
LeftAxis.LabelsFont.Color = clBlue
LeftAxis.LabelsFont.Height = -13
LeftAxis.LabelsFont.Name = 'Arial'
LeftAxis.LabelsFont.Style = [fsBold]
LeftAxis.Title.Caption = 'Sales'
LeftAxis.Title.Font.Charset = DEFAULT_CHARSET
LeftAxis.Title.Font.Color = clGray
LeftAxis.Title.Font.Height = -15
LeftAxis.Title.Font.Name = 'Arial'
LeftAxis.Title.Font.Style = [fsBold]
Legend.Visible = False

DEFAULT_CHARSET
LeftAxis.Title.Font.Color = clGreen
LeftAxis.Title.Font.Height = -15
LeftAxis.Title.Font.Name = 'Arial'
LeftAxis.Title.Font.Style = [fsBold, fsItalic]
Legend.Alignment = laTop
Legend.Color = clGray
Legend.ColorWidth = 35
Legend.Font.Charset = DEFAULT_CHARSET
Legend.Font.Color = clWhite
Legend.Font.Height = -12
Legend.Font.Name = 'Arial'
Legend.Font.Style = [fsItalic]
Legend.Frame.Color = clBlue
Legend.Frame.Width = 3
Legend.Frame.Visible = False
Legend.ShadowSize = 4
Legend.TopPos = 34
RightAxis.Grid.Visible = False
RightAxis.LabelsFont.Charset = DEFAULT_CHARSET
RightAxis.LabelsFont.Color = clBlue
RightAxis.LabelsFont.Height = -13
RightAxis.LabelsFont.Name = 'Arial'
RightAxis.LabelsFont.Style = [fsItalic]
RightAxis.Title.Caption = 'Red Area'
RightAxis.Title.Font.Charset = DEFAULT_CHARSET
RightAxis.Title.Font.Color = clRed
RightAxis.Title.Font.Height = -15
RightAxis.Title.Font.Name = 'Arial'
RightAxis.Title.Font.Style = [fsBold, fsItalic]
Color = clWhite
TabOrder = 4
<-------> object South: TAreaSeries
Marks.ArrowLength = 8
Marks.Frame.Color = 8454143

Foot.Font.Charset = DEFAULT_CHARSET
Foot.Font.Color = clRed
Foot.Font.Height = -12
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsItalic]
Foot.Frame.Color = clScrollBar
Foot.Text.Strings = (
'')
Gradient.Direction = gdLeftRight
Gradient.EndColor = clWhite
Gradient.StartColor = clYellow
LeftWall.Color = clSilver
Title.Font.Charset = DEFAULT_CHARSET
Title.Font.Color = clRed
Title.Font.Height = -16
Title.Font.Name = 'Arial'
Title.Font.Style = [fsBold, fsItalic]
Title.Frame.Color = clScrollBar
Title.Text.Strings = (
'Area Series')
BackColor = clSilver
BottomAxis.DateTimeFormat = 'd/mm'
BottomAxis.LabelsAngle = 90
BottomAxis.LabelsFont.Charset = DEFAULT_CHARSET
BottomAxis.LabelsFont.Color = clTeal
BottomAxis.LabelsFont.Height = -12
BottomAxis.LabelsFont.Name = 'Arial'
BottomAxis.LabelsFont.Style = []
BottomAxis.Title.Font.Charset = DEFAULT_CHARSET
BottomAxis.Title.Font.Color = clGreen
BottomAxis.Title.Font.Height = -16
BottomAxis.Title.Font.Name = 'Arial'
BottomAxis.Title.Font.Style = [fsItalic]
Chart3DPercent = 45
LeftAxis.Title.Caption = 'Green Area'
LeftAxis.Title.Font.Charset =

```
    Height = 173
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Color = 16777088
    BackWall.Pen.Color = clBlue
    BottomWall.Pen.Color = clBlue
    Foot.Font.Charset =
DEFAULT_CHARSET
    Foot.Font.Color = clYellow
    Foot.Font.Height = -11
    Foot.Font.Name = 'Arial'
    Foot.Font.Style = []
    Foot.Frame.Color = clScrollBar
    Foot.Text.Strings = (
      'Last values')
    Gradient.Direction =
gdBottomTop
    LeftWall.Color = clWhite
    LeftWall.Pen.Color = clBlue
    Title.Font.Charset =
DEFAULT_CHARSET
    Title.Font.Color = clPurple
    Title.Font.Height = -16
    Title.Font.Name = 'Arial'
    Title.Font.Style = [fsBold, fsItalic]
    Title.Frame.Color = clScrollBar
    Title.Text.Strings = (
      'Point Series')
    BackColor = 16777088
    BottomAxis.Axis.Color = clBlue
    BottomAxis.Axis.Width = 1
    BottomAxis.DateTimeFormat =
'd/MM/yy'
    BottomAxis.Grid.Color = clBlue
    BottomAxis.LabelsFont.Charset =
DEFAULT_CHARSET
    BottomAxis.LabelsFont.Color =
clGreen
    BottomAxis.LabelsFont.Height = -
12
    BottomAxis.LabelsFont.Name =
'Arial'
    BottomAxis.LabelsFont.Style = []
    BottomAxis.LabelsSeparation = 0
    BottomAxis.MinorTickCount = 4
    BottomAxis.TickLength = 3
    BottomAxis.Ticks.Color = clWhite
    BottomAxis.Title.Font.Charset =
DEFAULT_CHARSET
```

```
      Marks.Visible = False
      SeriesColor = clGreen
      DrawArea = True
      Pointer.InflateMargins = True
      Pointer.Style = psRectangle
      Pointer.Visible = False
      XValues.DateTime = True
      XValues.Name = 'X'
      XValues.Multiplier =
1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier =
1.000000000000000000
      YValues.Order = loNone
      Left = 441
      Top = 206
    end
    <-------> object North:
TAreaSeries
      Marks.ArrowLength = 8
      Marks.Frame.Color = 8454143
      Marks.Visible = False
      SeriesColor = clRed
      VertAxis = aRightAxis
      DrawArea = True
      Pointer.InflateMargins = True
      Pointer.Style = psRectangle
      Pointer.Visible = False
      XValues.DateTime = True
      XValues.Name = 'X'
      XValues.Multiplier =
1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier =
1.000000000000000000
      YValues.Order = loNone
      Left = 441
      Top = 173
    end
  end
  <-------> object Chart4: TChart
    Left = 466
    Top = 256
    Width = 311
```

13
RightAxis.LabelsFont.Name = 'Arial'
RightAxis.LabelsFont.Style = []
RightAxis.Visible = False
TopAxis.Visible = False
Color = 8454143
TabOrder = 5
<-------> object Speaking: TPointSeries
Marks.ArrowLength = 8
Marks.Frame.Color = 8454143
Marks.Visible = False
SeriesColor = clRed
VertAxis = aRightAxis
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = True
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 503
Top = 301
end
<-------> object Writing: TPointSeries
Marks.ArrowLength = 8
Marks.Frame.Color = 8454143
Marks.Visible = False
SeriesColor = clGreen
Pointer.Brush.Color = clGreen
Pointer.InflateMargins = True
Pointer.Pen.Color = 64
Pointer.Style = psCircle
Pointer.Visible = True
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False

BottomAxis.Title.Font.Color = clGreen
BottomAxis.Title.Font.Height = -16
BottomAxis.Title.Font.Name = 'Arial'
BottomAxis.Title.Font.Style = [fsItalic]
Chart3DPercent = 20
Frame.Color = clBlue
LeftAxis.Axis.Color = clBlue
LeftAxis.Axis.Width = 1
LeftAxis.Grid.Color = clBlue
LeftAxis.LabelsFont.Charset = DEFAULT_CHARSET
LeftAxis.LabelsFont.Color = clTeal
LeftAxis.LabelsFont.Height = -13
LeftAxis.LabelsFont.Name = 'Arial'
LeftAxis.LabelsFont.Style = [fsItalic]
LeftAxis.Ticks.Color = clFuchsia
LeftAxis.Title.Font.Charset = DEFAULT_CHARSET
LeftAxis.Title.Font.Color = clNavy
LeftAxis.Title.Font.Height = -15
LeftAxis.Title.Font.Name = 'Arial'
LeftAxis.Title.Font.Style = [fsBold]
LeftAxis.Visible = False
Legend.Alignment = laBottom
Legend.ColorWidth = 50
Legend.Font.Charset = DEFAULT_CHARSET
Legend.Font.Color = clPurple
Legend.Font.Height = -11
Legend.Font.Name = 'Arial'
Legend.Font.Style = []
Legend.LegendStyle = lsLastValues
Legend.TopPos = 7
RightAxis.Grid.Color = clLime
RightAxis.LabelsFont.Charset = DEFAULT_CHARSET
RightAxis.LabelsFont.Color = clYellow
RightAxis.LabelsFont.Height = -

```
    Font.Color = clBlack
    Font.Height = -16
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 0
    OnClick = CheckBox1Click
  end
  <-------> object BitBtn1: TBitBtn
    Left = 8
    Top = 160
    Width = 97
    Height = 29
    Caption = '    &Print all'
    TabOrder = 1
    OnClick = BitBtn1Click
    Glyph.Data = {
    NumGlyphs = 2
  end
  <-------> object RadioGroup1:
TRadioGroup
    Left = 4
    Top = 96
    Width = 109
    Height = 53
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -12
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ItemIndex = 0
    Items.Strings = (
      'Proportional'
      'More resolution')
    ParentFont = False
    TabOrder = 2
  end
  <-------> object CheckBox2:
TCheckBox
    Left = 12
    Top = 20
    Width = 41
    Height = 17
    Caption = '3&D'
    Checked = True
    State = cbChecked
    TabOrder = 3
```

```
    YValues.Name = 'Y'
    YValues.Multiplier =
1.000000000000000000
    YValues.Order = loNone
    Left = 501
    Top = 267
  end
  <-------> object Reading:
TPointSeries
    Marks.ArrowLength = 8
    Marks.Frame.Color = 8454143
    Marks.Visible = False
    SeriesColor = clYellow
    Pointer.Brush.Color = clYellow
    Pointer.HorizSize = 5
    Pointer.InflateMargins = True
    Pointer.Style = psTriangle
    Pointer.VertSize = 5
    Pointer.Visible = True
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier =
1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier =
1.000000000000000000
    YValues.Order = loNone
    Left = 499
    Top = 233
  end
 end
 <-------> object Panel1: TPanel
  Left = -1
  Top = 40
  Width = 121
  Height = 417
  TabOrder = 1
  <-------> object CheckBox1:
TCheckBox
    Left = 12
    Top = 52
    Width = 105
    Height = 17
    Caption = '&Animate !!!'
    Font.Charset =
DEFAULT_CHARSET
```

```
    ParentFont = False
  end
 end
 <-------> object Timer1: TTimer
  Enabled = False
  Interval = 1
  OnTimer = Timer1Timer
  Left = 102
  Top = 254
 end
 <-------> object SaveDialog1: TSaveDialog
  DefaultExt = 'BMP'
  FileName = 'TEEDEMO.BMP'
  Filter = 'Bitmap Files|*.bmp'
  Options = [ofOverwritePrompt, ofHideReadOnly]
  Title = 'TeeChart DEMO Save to Bitmap'
  Left = 162
  Top = 444
 end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
<-------> object BitmapForm: TBitmapForm
 Left = 183
 Top = 118
 Width = 675
 Height = 421
 Caption = 'TeeChart Background Image demo'
 Color = clBtnFace
 ParentFont = True
 OldCreateOrder = True
 Position = poScreenCenter
 OnCreate = FormCreate
 PixelsPerInch = 96
 TextHeight = 13
 <-------> object Chart1: TChart
  Left = 145
  Top = 0
  Width = 522
  Height = 387
  BackImage.Data = {

07544269746D6170E62B0000424DE
62B000000000000760000002800000O
    OnClick = CheckBox2Click
  end
  <-------> object BitBtn2: TBitBtn
   Left = 8
   Top = 264
   Width = 97
   Height = 29
   Caption = '    &Save all '
   TabOrder = 4
   OnClick = BitBtn2Click
   Glyph.Data = {
   NumGlyphs = 2
  end
  <-------> object BitBtn3: TBitBtn
   Left = 8
   Top = 368
   Width = 100
   Height = 37
   Caption = '    Close'
   TabOrder = 5
   Kind = bkClose
  end
 end
 <-------> object Panel2: TPanel
  Left = 0
  Top = 0
  Width = 777
  Height = 40
  Align = alTop
  TabOrder = 2
  <-------> object Label1: TLabel
   Left = 12
   Top = 12
   Width = 471
   Height = 16
   HelpType = htKeyword
   Alignment = taCenter
   Caption =
    'Left drag (Up/Down) to Zoom. Right drag to Scroll. Invert Zoom r' +
    'ectangle to reset.'
   Font.Charset = ANSI_CHARSET
   Font.Color = clNavy
   Font.Height = -13
   Font.Name = 'Arial'
   Font.Style = [fsItalic]
```

```
  Left = 0
  Top = 0
  Width = 145
  Height = 387
  Align = alLeft
  TabOrder = 1
  <-------> object RadioGroup1:
TRadioGroup
    Left = 22
    Top = 12
    Width = 91
    Height = 85
    Caption = 'Bitmap Mode:'
    ItemIndex = 1
    Items.Strings = (
      'Stretch'
      'Tile'
      'Center')
    TabOrder = 0
    OnClick = RadioGroup1Click
  end
  <-------> object BitBtn1: TBitBtn
    Left = 12
    Top = 132
    Width = 125
    Height = 25
    Caption = '&Open Bitmap...'
    TabOrder = 1
    OnClick = BitBtn1Click
    Glyph.Data = {
    NumGlyphs = 2
  end
  <-------> object BitBtn3: TBitBtn
    Left = 28
    Top = 348
    Width = 85
    Height = 37
    TabOrder = 2
    Kind = bkClose
  end
  <-------> object CheckBox1:
TCheckBox
    Left = 36
    Top = 104
    Width = 65
    Height = 17
    Caption = 'Inside'
    Checked = True
```

```
9900
    BackImageInside = True
    BackImageMode = pbmTile
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    Foot.Font.Charset =
DEFAULT_CHARSET
    Foot.Font.Color = clRed
    Foot.Font.Height = -12
    Foot.Font.Name = 'Arial'
    Foot.Font.Style = [fsItalic]
    Foot.Frame.Color = clScrollBar
    Title.Brush.Color = clBlack
    Title.Color = clBlack
    Title.Font.Charset =
DEFAULT_CHARSET
    Title.Font.Color = clWhite
    Title.Font.Height = -21
    Title.Font.Name = 'Arial'
    Title.Font.Style = [fsBold, fsItalic]
    Title.Frame.Color = clScrollBar
    Title.Text.Strings = (
      'TeeChart Panel Bitmap
Example')
    Chart3DPercent = 45
    Legend.TopPos = 20
    Align = alClient
    BevelInner = bvRaised
    BorderWidth = 28
    TabOrder = 0
    <-------> object Series1:
TBarSeries
      Marks.ArrowLength = 20
      Marks.Visible = True
      SeriesColor = clRed
      XValues.DateTime = True
      XValues.Name = 'X'
      XValues.Multiplier =
1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Bar'
      YValues.Multiplier =
1.000000000000000000
      YValues.Order = loNone
    end
  end
  <-------> object Panel1: TPanel
```

```
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Panel1: TPanel
    Left = 0
    Top = 0
    Width = 153
    Height = 355
    Align = alLeft
    TabOrder = 0
    <-------> object CheckBox1:
TCheckBox
      Left = 20
      Top = 13
      Width = 125
      Height = 17
      Caption = 'Animate !!!'
      Font.Charset =
DEFAULT_CHARSET
      Font.Color = clBlack
      Font.Height = -19
      Font.Name = 'Arial'
      Font.Style = [fsBold]
      ParentFont = False
      TabOrder = 0
      OnClick = CheckBox1Click
    end
    <-------> object BitBtn3: TBitBtn
      Left = 32
      Top = 309
      Width = 89
      Height = 33
      TabOrder = 1
      Kind = bkClose
    end
    <-------> object Memo1: TMemo
      Left = 16
      Top = 48
      Width = 121
      Height = 137
      Lines.Strings = (
        'Each Arrow is '
        'represented as a Series '
        'point with Starting and '
        'Ending coordinates.'
      State = cbChecked
      TabOrder = 3
      OnClick = CheckBox1Click
    end
    <-------> object Button1: TButton
      Left = 12
      Top = 172
      Width = 125
      Height = 25
      Caption = '&Clear'
      TabOrder = 4
      OnClick = Button1Click
    end
    <-------> object Memo1: TMemo
      Left = 13
      Top = 216
      Width = 121
      Height = 105
      Lines.Strings = (
        'Images can be used as '
        'backgrounds in '
        'centered,'
        'tiled or stretched mode.')
      TabOrder = 5
    end
  end
  <-------> object OpenDialog1:
TOpenDialog
    DefaultExt = 'BMP'
    Filter = 'Bitmap Files|*.BMP'
    Options = [ofReadOnly,
ofPathMustExist, ofFileMustExist]
    Title = 'Select a Bitmap File'
    Left = 100
    Top = 43
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
<-------> object ArrowsForm:
TArrowsForm
  Left = 188
  Top = 111
  Width = 600
  Height = 389
  Caption = 'TeeChart Arrow Series
Example'
  Color = clBtnFace
```

```
    Pointer.HorizSize = 32
    Pointer.InflateMargins = False
    Pointer.Style = psRectangle
    Pointer.VertSize = 24
    Pointer.Visible = True
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier =
1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier =
1.000000000000000000
    YValues.Order = loNone
    EndXValues.DateTime = False
    EndXValues.Name = 'EndX'
    EndXValues.Multiplier =
1.000000000000000000
    EndXValues.Order = loNone
    EndYValues.DateTime = False
    EndYValues.Name = 'EndY'
    EndYValues.Multiplier =
1.000000000000000000
    EndYValues.Order = loNone
    StartXValues.DateTime = True
    StartXValues.Name = 'X'
    StartXValues.Multiplier =
1.000000000000000000
    StartXValues.Order =
loAscending
    StartYValues.DateTime = False
    StartYValues.Name = 'Y'
    StartYValues.Multiplier =
1.000000000000000000
    StartYValues.Order = loNone
    Left = 104
    Top = 195
   end
  end
  <-------> object Timer1: TTimer
   Enabled = False
   Interval = 1
   OnTimer = Timer1Timer
   Left = 62
   Top = 195
  end
End
```

```
     ''
     'This demo changes '
     'arrow positions randomly '
     'using a'
     'TTimer component.')
    TabOrder = 2
   end
  end
  <-------> object Chart1: TChart
   Left = 153
   Top = 0
   Width = 439
   Height = 355
   AnimatedZoom = True
   BackWall.Brush.Color = clWhite
   BackWall.Brush.Style = bsClear
   BackWall.Color = 8454143
   BottomWall.Color = 8454016
   Foot.Font.Charset =
DEFAULT_CHARSET
   Foot.Font.Color = clRed
   Foot.Font.Height = -12
   Foot.Font.Name = 'Arial'
   Foot.Font.Style = [fsItalic]
   Foot.Frame.Color = clScrollBar
   LeftWall.Color = 16777088
   Title.Frame.Color = clScrollBar
   Title.Text.Strings = (
    '')
   Title.Visible = False
   BackColor = 8454143
   Chart3DPercent = 50
   Legend.Visible = False
   RightAxis.Grid.Visible = False
   RightAxis.MinorTickLength = 3
   RightAxis.TickLength = 5
   Align = alClient
   TabOrder = 1
   <-------> object ArrowSeries1:
TArrowSeries
    ColorEachPoint = True
    HorizAxis = aTopAxis
    Marks.ArrowLength = 8
    Marks.Frame.Visible = False
    Marks.Transparent = True
    Marks.Visible = False
    SeriesColor = clRed
    VertAxis = aRightAxis
```

'maStacked'
'maStacked100')
TabOrder = 0
OnClick = RadioGroup1Click
end
<-------> object BitBtn1: TBitBtn
Left = 18
Top = 350
Width = 89
Height = 33
TabOrder = 1
Kind = bkClose
end
<-------> object CheckBox1: TCheckBox
Left = 14
Top = 34
Width = 97
Height = 17
Caption = '3D'
Checked = True
State = cbChecked
TabOrder = 2
OnClick = CheckBox1Click
end
<-------> object CheckBox2: TCheckBox
Left = 14
Top = 9
Width = 97
Height = 17
Caption = '&Animate !!!'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -16
Font.Name = 'Arial'
Font.Style = [fsBold]
ParentFont = False
TabOrder = 3
OnClick = CheckBox2Click
end
end
<-------> object Chart1: TChart
Left = 145
Top = 0
Width = 445
Height = 399

\\\\\\\\\\\\\\\\\\\\\\\\\\\\
<-------> object AreasForm: TAreasForm
Left = 192
Top = 110
Width = 598
Height = 433
ActiveControl = Chart1
Caption = 'Area Series Example'
Color = clBtnFace
ParentFont = True
OldCreateOrder = True
Position = poScreenCenter
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> object Panel1: TPanel
Left = 0
Top = 0
Width = 145
Height = 399
Align = alLeft
TabOrder = 0
<-------> object Label1: TLabel
Left = 14
Top = 196
Width = 62
Height = 22
HelpType = htKeyword
Caption = 'Label1'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -19
Font.Name = 'Arial'
Font.Style = [fsBold, fsItalic]
ParentFont = False
end
<-------> object RadioGroup1: TRadioGroup
Left = 12
Top = 59
Width = 117
Height = 122
Caption = 'Area Series Mode:'
ItemIndex = 0
Items.Strings = (
'maNone'

end
<-------> object AreaSeries2: TAreaSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clGreen
DrawArea = True
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 227
Top = 36
end
<-------> object AreaSeries3: TAreaSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clYellow
DrawArea = True
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 265
Top = 34
end
end
<-------> object Timer1: TTimer
Enabled = False

BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
Foot.Font.Charset = DEFAULT_CHARSET
Foot.Font.Color = clRed
Foot.Font.Height = -12
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsItalic]
Title.Alignment = taLeftJustify
Title.Font.Charset = DEFAULT_CHARSET
Title.Font.Color = clBlue
Title.Font.Height = -13
Title.Font.Name = 'Arial'
Title.Font.Style = [fsBold]
Title.Text.Strings = (
'Area Series Example.'
'Move mouse over Areas.')
BottomAxis.LabelsAngle = 90
Chart3DPercent = 50
Legend.Alignment = laBottom
Align = alClient
TabOrder = 1
OnMouseMove = Chart1MouseMove
<-------> object AreaSeries1: TAreaSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clRed
AreaLinesPen.Color = clAqua
AreaLinesPen.Style = psDot
DrawArea = True
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 189
Top = 36

end
<-------> object Label2: TLabel
Left = 168
Top = 353
Width = 300
Height = 13
HelpType = htKeyword
Caption = '© Copyright 1996-2001 by David Berneda. All Rights Reserved.'
Color = clNavy
Font.Charset = DEFAULT_CHARSET
Font.Color = clWhite
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentColor = False
ParentFont = False
end
<-------> object Panel1: TPanel
Left = 0
Top = 376
Width = 632
Height = 30
Align = alBottom
Alignment = taLeftJustify
TabOrder = 0
<-------> object Label11: TLabel
Left = 12
Top = 49
Width = 3
Height = 13
HelpType = htKeyword
end
<-------> object Label13: TLabel
Left = 12
Top = 6
Width = 129
Height = 19
HelpType = htKeyword
Caption = 'Steema Software'
Font.Charset = DEFAULT_CHARSET
Font.Color = clNavy
Font.Height = -16
Font.Name = 'Arial'
Font.Style = [fsBold]

Interval = 1
OnTimer = Timer1Timer
Left = 88
Top = 91
end
End
\\
<-------> object TeeMainForm: TTeeMainForm
Left = 225
Top = 188
ActiveControl = Button2
BorderIcons = [biSystemMenu]
BorderStyle = bsDialog
Caption = 'TeeChart Standard version 4.0 - Demo'
ClientHeight = 406
ClientWidth = 632
Color = clNavy
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> object Image1: TImage
Left = 484
Top = 26
Width = 141
Height = 45
HelpType = htKeyword
AutoSize = True
Picture.Data = {
07544269746D6170861D0000424D861D000000000000360400002800000008D00
00002D0000000100080000000000501900000000000000000000000100000001
OnClick = Label10Click

end
<-------> object Button1: TButton
Left = 536
Top = 4
Width = 89
Height = 23
Caption = 'Close'
ModalResult = 1
TabOrder = 0
OnClick = Button1Click
end
end
<-------> object TPanel
Left = 476
Top = 102
Width = 157
Height = 251
BevelOuter = bvNone
BorderWidth = 1
Color = clNavy
TabOrder = 1
<-------> object Button2: TButton
Left = 8
Top = -1
Width = 141
Height = 41
Caption = 'Show me the Demo !'
Default = True
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -12
Font.Name = 'Arial'
Font.Style = [fsBold]
ParentFont = False
TabOrder = 0
OnClick = Button2Click
end
<-------> object Button3: TButton
Left = 8
Top = 85
Width = 141
Height = 29
Caption = 'Features...'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -12
ParentFont = False
end
<-------> object Label15: TLabel
Left = 212
Top = 9
Width = 27
Height = 13
HelpType = htKeyword
Caption = 'email:'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
<-------> object Label16: TLabel
Left = 244
Top = 9
Width = 85
Height = 13
HelpType = htKeyword
Caption = 'info@steema.com'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
end
<-------> object Label3: TLabel
Left = 375
Top = 6
Width = 119
Height = 16
Cursor = crHandPoint
HelpType = htKeyword
Caption = 'www.steema.com'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlue
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold, fsUnderline]
ParentFont = False
OnClick = Label3Click

Title.Visible = False
AxisVisible = False
ClipPoints = False
Frame.Visible = False
Legend.Color = clSilver
Legend.Visible = False
View3DWalls = False
BevelWidth = 2
Color = clSilver
TabOrder = 4
OnMouseDown = Chart2MouseDown
OnMouseUp = Chart2MouseUp
<-------> object Label12: TLabel
Left = 25
Top = 16
Width = 97
Height = 32
HelpType = htKeyword
Caption = 'About...'
Font.Charset = DEFAULT_CHARSET
Font.Color = clRed
Font.Height = -29
Font.Name = 'Times New Roman'
Font.Style = [fsBold]
ParentFont = False
Transparent = True
OnClick = Label10Click
OnMouseDown = Chart2MouseDown
end
<-------> object FastLineSeries1: TFastLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clAqua
LinePen.Color = clAqua
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone

Font.Name = 'Arial'
Font.Style = [fsBold]
ParentFont = False
TabOrder = 1
OnClick = Button3Click
end
<-------> object Button4: TButton
Left = 8
Top = 122
Width = 141
Height = 29
Caption = 'Pro Specifications...'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -12
Font.Name = 'Arial'
Font.Style = [fsBold]
ParentFont = False
TabOrder = 2
OnClick = Button4Click
end
<-------> object Button6: TButton
Left = 8
Top = 48
Width = 141
Height = 29
Caption = 'Chart Styles...'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -12
Font.Name = 'Arial'
Font.Style = [fsBold]
ParentFont = False
TabOrder = 3
OnClick = Button6Click
end
<-------> object Chart2: TChart
Left = 8
Top = 168
Width = 141
Height = 73
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
BackWall.Pen.Visible = False
Title.Text.Strings = (
'TChart')

```
    Title.Text.Strings = (
      '')
    AxisVisible = False
    Chart3DPercent = 50
    ClipPoints = False
    Frame.Visible = False
    Legend.Visible = False
    View3DOptions.Elevation = 315
    View3DOptions.Orthogonal =
False
    View3DOptions.Perspective = 0
    View3DOptions.Rotation = 360
    View3DOptions.Zoom = 92
    View3DWalls = False
    BevelOuter = bvNone
    Color = clAqua
    TabOrder = 1
    <-------> object PieSeries6:
TPieSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clRed
      CustomYRadius = 25
      ExplodeBiggest = 25
      OtherSlice.Text = 'Other'
      PieValues.DateTime = False
      PieValues.Name = 'Pie'
      PieValues.Multiplier =
1.000000000000000000
      PieValues.Order = loNone
    end
  end
  <-------> object Chart8: TChart
    Left = 3
    Top = 163
    Width = 97
    Height = 66
    AllowPanning = pmNone
    AllowZoom = False
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    MarginBottom = 10
    MarginLeft = 5
    MarginRight = 5
    MarginTop = 0
    Title.Font.Charset =
DEFAULT_CHARSET
    end
    <-------> object FastLineSeries2:
TFastLineSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clYellow
      LinePen.Color = clYellow
      XValues.DateTime = True
      XValues.Name = 'X'
      XValues.Multiplier =
1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier =
1.000000000000000000
      YValues.Order = loNone
    end
  end
end
<-------> object Panel2: TPanel
  Left = 9
  Top = 102
  Width = 465
  Height = 241
  BevelWidth = 2
  Color = clAqua
  TabOrder = 2
  <-------> object Chart7: TChart
    Left = 3
    Top = 7
    Width = 97
    Height = 66
    AllowPanning = pmNone
    AllowZoom = False
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    MarginBottom = 10
    MarginLeft = 5
    MarginRight = 5
    MarginTop = 0
    Title.Font.Charset =
DEFAULT_CHARSET
    Title.Font.Color = clBlack
    Title.Font.Height = -17
    Title.Font.Name = 'Arial'
    Title.Font.Style = [fsBold]
```

```
    MarginRight = 5
    MarginTop = 0
    Title.Font.Charset =
DEFAULT_CHARSET
    Title.Font.Color = clBlack
    Title.Font.Height = -17
    Title.Font.Name = 'Arial'
    Title.Font.Style = [fsBold]
    Title.Text.Strings = (
      '')
    AxisVisible = False
    Chart3DPercent = 100
    ClipPoints = False
    Frame.Visible = False
    Legend.Visible = False
    View3DOptions.Elevation = 315
    View3DOptions.Orthogonal =
False
    View3DOptions.Perspective = 0
    View3DOptions.Rotation = 360
    View3DOptions.Zoom = 92
    View3DWalls = False
    BevelOuter = bvNone
    Color = clAqua
    TabOrder = 4
    <-------> object PieSeries9:
TPieSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clRed
      CustomYRadius = 25
      ExplodeBiggest = 40
      OtherSlice.Text = 'Other'
      PieValues.DateTime = False
      PieValues.Name = 'Pie'
      PieValues.Multiplier =
1.000000000000000000
      PieValues.Order = loNone
    end
  end
  <-------> object Chart9: TChart
    Left = 206
    Top = 4
    Width = 255
    Height = 221
    AllowPanning = pmNone
    AllowZoom = False
    BackWall.Brush.Color = clWhite
    Title.Font.Color = clBlack
    Title.Font.Height = -17
    Title.Font.Name = 'Arial'
    Title.Font.Style = [fsBold]
    Title.Text.Strings = (
      '')
    AxisVisible = False
    Chart3DPercent = 95
    ClipPoints = False
    Frame.Visible = False
    Legend.Visible = False
    View3DOptions.Elevation = 315
    View3DOptions.Orthogonal =
False
    View3DOptions.Perspective = 0
    View3DOptions.Rotation = 360
    View3DOptions.Zoom = 87
    View3DWalls = False
    BevelOuter = bvNone
    Color = clAqua
    TabOrder = 2
    <-------> object PieSeries7:
TPieSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clRed
      CustomYRadius = 25
      ExplodeBiggest = 20
      OtherSlice.Text = 'Other'
      PieValues.DateTime = False
      PieValues.Name = 'Pie'
      PieValues.Multiplier =
1.000000000000000000
      PieValues.Order = loNone
    end
  end
  <-------> object Chart10: TChart
    Left = 105
    Top = 139
    Width = 97
    Height = 66
    AllowPanning = pmNone
    AllowZoom = False
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    MarginBottom = 10
    MarginLeft = 5
```

```
    OtherSlice.Text = 'Other'
    PiePen.Visible = False
    PieValues.DateTime = False
    PieValues.Name = 'Pie'
    PieValues.Multiplier = 
1.000000000000000000
    PieValues.Order = loNone
    DataSources = (
      'Series1'
      'PieSeries6'
      'PieSeries7'
      'PieSeries9')
    <-------> object 
PieSeries8TAddTeeFunction: 
TAddTeeFunction
    end
   end
  end
  <-------> object Chart1: TChart
   Left = 105
   Top = 31
   Width = 97
   Height = 66
   AllowPanning = pmNone
   AllowZoom = False
   BackWall.Brush.Color = clWhite
   BackWall.Brush.Style = bsClear
   BackWall.Pen.Visible = False
   MarginBottom = 10
   MarginLeft = 5
   MarginRight = 5
   MarginTop = 0
   Title.Font.Charset = 
DEFAULT_CHARSET
   Title.Font.Color = clBlack
   Title.Font.Height = -17
   Title.Font.Name = 'Arial'
   Title.Font.Style = [fsBold]
   Title.Text.Strings = (
     '')
   AxisVisible = False
   Chart3DPercent = 50
   ClipPoints = False
   Frame.Visible = False
   Legend.Visible = False
   View3DOptions.Elevation = 315
   View3DOptions.Orthogonal = 
False
```

```
   BackWall.Brush.Style = bsClear
   BackWall.Pen.Visible = False
   MarginBottom = 10
   MarginLeft = 5
   MarginRight = 5
   MarginTop = 0
   Title.Font.Charset = 
DEFAULT_CHARSET
   Title.Font.Color = clBlack
   Title.Font.Height = -24
   Title.Font.Name = 'Arial'
   Title.Font.Style = [fsBold]
   Title.Text.Strings = (
     '')
   Title.Visible = False
   AxisVisible = False
   Chart3DPercent = 70
   ClipPoints = False
   Frame.Visible = False
   Legend.TextStyle = ltsPlain
   Legend.Visible = False
   View3DOptions.Elevation = 315
   View3DOptions.Orthogonal = 
False
   View3DOptions.Perspective = 0
   View3DOptions.Rotation = 360
   View3DWalls = False
   BevelOuter = bvNone
   Color = clAqua
   TabOrder = 3
   <-------> object PieSeries8: 
TPieSeries
    Marks.Arrow.Color = clBlack
    Marks.ArrowLength = 1
    Marks.Font.Charset = 
DEFAULT_CHARSET
    Marks.Font.Color = clNavy
    Marks.Font.Height = -11
    Marks.Font.Name = 'Arial'
    Marks.Font.Style = []
    Marks.Frame.Visible = False
    Marks.Transparent = True
    Marks.Visible = True
    DataSource = Series1
    PercentFormat = '##0.00 %'
    SeriesColor = clRed
    CustomYRadius = 35
    ExplodeBiggest = 40
```

```
    Alignment = taCenter
    AutoSize = False
    Caption = 'The 100% Native Charting VCL / CLX Library'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlue
    Font.Height = -19
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
  end
  <-------> object Label9: TLabel
    Left = 8
    Top = 44
    Width = 449
    Height = 19
    HelpType = htKeyword
    Alignment = taCenter
    AutoSize = False
    Caption = 'For Borland Delphi, Kylix and C++ Builder'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -16
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
  end
 end
 <-------> object Timer1: TTimer
  Interval = 1
  OnTimer = Timer1Timer
  Left = 408
  Top = 283
 end
 <-------> object Timer2: TTimer
  Interval = 200
  OnTimer = Timer2Timer
  Left = 356
  Top = 281
 end
End

\\\\\\\\\\\\\\\\\\\\\\\\
<-------> object MetafileForm: TMetafileForm
```

```
    View3DOptions.Perspective = 0
    View3DOptions.Rotation = 360
    View3DWalls = False
    BevelOuter = bvNone
    Color = clAqua
    TabOrder = 0
    <-------> object Series1: TPieSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clRed
      CustomYRadius = 25
      ExplodeBiggest = 25
      OtherSlice.Text = 'Other'
      PieValues.DateTime = False
      PieValues.Name = 'Pie'
      PieValues.Multiplier = 1.000000000000000000
      PieValues.Order = loNone
    end
   end
  end
  <-------> object TPanel
   Left = 8
   Top = 359
   Width = 156
   Height = 1
   TabOrder = 3
  end
  <-------> object TPanel
   Left = 475
   Top = 360
   Width = 153
   Height = 1
   TabOrder = 4
  end
  <-------> object Panel3: TPanel
   Left = 8
   Top = 11
   Width = 465
   Height = 81
   TabOrder = 5
   <-------> object Label1: TLabel
    Left = 8
    Top = 15
    Width = 449
    Height = 22
    HelpType = htKeyword
```

Align = alTop
TabOrder = 0
<-------> object BarSeries1: TBarSeries
ColorEachPoint = True
Marks.ArrowLength = 20
Marks.Visible = True
SeriesColor = clRed
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 15
Top = 52
end
end
<-------> object Panel1: TPanel
Left = 0
Top = 0
Width = 546
Height = 41
Align = alTop
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'Arial'
Font.Style = []
ParentFont = False
TabOrder = 1
<-------> object BitBtn2: TBitBtn
Left = 12
Top = 4
Width = 149
Height = 33
Caption = '&Save to *.WMF...'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []

Left = 199
Top = 116
Width = 554
Height = 443
Caption = 'TeeChart -- Metafile Format Demo'
Color = clBtnFace
ParentFont = True
OldCreateOrder = True
Position = poScreenCenter
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> object Image1: TImage
Left = 0
Top = 258
Width = 546
Height = 151
HelpType = htKeyword
Align = alClient
Stretch = True
end
<-------> object Chart1: TChart
Left = 0
Top = 41
Width = 546
Height = 176
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
Foot.Font.Charset = DEFAULT_CHARSET
Foot.Font.Color = clRed
Foot.Font.Height = -12
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsItalic]
Foot.Frame.Color = clScrollBar
Title.Frame.Color = clWhite
Title.Text.Strings = (
'TChart Component')
BottomAxis.DateTimeFormat = 'MM/dd/yy'
BottomAxis.Grid.Color = clScrollBar
Chart3DPercent = 35
LeftAxis.Grid.Color = clScrollBar
RightAxis.Grid.Color = clScrollBar
TopAxis.Grid.Color = clScrollBar

```
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
<-------> object PagesForm: TPagesForm
  Left = 203
  Top = 106
  Width = 608
  Height = 413
  Caption = 'TeeChart Sample.  How to divide Charts in several Pages.'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 194
    Top = 0
    Width = 406
    Height = 379
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    Foot.Frame.Color = clScrollBar
    Title.Frame.Color = clScrollBar
    Title.Text.Strings = (
      'TChart Component')
    BottomAxis.Grid.Color = clScrollBar
    LeftAxis.Grid.Color = clScrollBar
    Legend.Alignment = laTop
    RightAxis.Grid.Color = clScrollBar
    ScaleLastPage = False
    TopAxis.Grid.Color = clScrollBar
    OnPageChange = Chart1PageChange
    Align = alClient
    TabOrder = 0
    <-------> object LineSeries1: TLineSeries
```

```
      ParentFont = False
      TabOrder = 0
      OnClick = BitBtn2Click
      Glyph.Data = {
      NumGlyphs = 2
    end
    <-------> object BitBtn4: TBitBtn
      Left = 358
      Top = 4
      Width = 89
      Height = 33
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlack
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 2
      Kind = bkClose
    end
  end
  <-------> object Panel2: TPanel
    Left = 0
    Top = 217
    Width = 546
    Height = 41
    Align = alTop
    Caption = 'This is a TImage Delphi Component with a Metafile Image'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 2
  end
  <-------> object SaveDialog1: TSaveDialog
    DefaultExt = 'WMF'
    FileName = 'metademo.wmf'
    Filter = 'Metafile Format|*.WMF'
    Options = [ofOverwritePrompt]
    Left = 196
    Top = 49
```

```
    Top = 160
  end
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 0
  Width = 194
  Height = 379
  Align = alLeft
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  ParentFont = False
  TabOrder = 1
  <-------> object Label2: TLabel
    Left = 11
    Top = 10
    Width = 104
    Height = 13
    HelpType = htKeyword
    Caption = '&Num Points Per
Page:'
    FocusControl = SpinEdit1
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
  end
  <-------> object Label1: TLabel
    Left = 11
    Top = 76
    Width = 62
    Height = 22
    HelpType = htKeyword
    Caption = 'Label1'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -19
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
```

```
    Marks.ArrowLength = 8
    Marks.Visible = False
    SeriesColor = clRed
    Pointer.InflateMargins = True
    Pointer.Style = psRectangle
    Pointer.Visible = False
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier =
1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier =
1.000000000000000000
    YValues.Order = loNone
    Left = 311
    Top = 156
  end
  <-------> object BubbleSeries1:
TBubbleSeries
    ColorEachPoint = False
    Marks.ArrowLength = 8
    Marks.Frame.Visible = False
    Marks.Transparent = True
    Marks.Visible = False
    SeriesColor = clGreen
    Pointer.HorizSize = 26
    Pointer.InflateMargins = True
    Pointer.Style = psCircle
    Pointer.VertSize = 26
    Pointer.Visible = True
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier =
1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier =
1.000000000000000000
    YValues.Order = loNone
    RadiusValues.DateTime = False
    RadiusValues.Name = 'Radius'
    RadiusValues.Multiplier =
1.000000000000000000
    RadiusValues.Order = loNone
    Left = 395
```

```
    Glyph.Data = {
    Layout = blGlyphRight
    NumGlyphs = 2
  end
  <-------> object BLastPage: TBitBtn
    Left = 102
    Top = 106
    Width = 79
    Height = 32
    Caption = '&Last'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 3
    OnClick = BLastPageClick
    Glyph.Data = {
    Layout = blGlyphRight
    NumGlyphs = 2
  end
  <-------> object BFirstPage: TBitBtn
    Left = 14
    Top = 106
    Width = 79
    Height = 32
    Caption = '&First'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 4
    OnClick = BFirstPageClick
    Glyph.Data = {
    NumGlyphs = 2
  end
  <-------> object CheckBox1: TCheckBox
    Left = 14
    Top = 44
    Width = 131
```

```
  end
  <-------> object SpinEdit1: TSpinEdit
    Left = 121
    Top = 7
    Width = 58
    Height = 22
    MaxValue = 100000
    MinValue = 0
    TabOrder = 0
    Value = 0
    OnChange = SpinEdit1Change
  end
  <-------> object ButtonPrevious: TBitBtn
    Left = 14
    Top = 149
    Width = 79
    Height = 32
    Caption = '&Previous'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 1
    OnClick = ButtonPreviousClick
    Glyph.Data = {
    NumGlyphs = 2
  end
  <-------> object ButtonNext: TBitBtn
    Left = 102
    Top = 149
    Width = 79
    Height = 32
    Caption = '&Next'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 2
    OnClick = ButtonNextClick
```

```
  Top = 108
  Width = 575
  Height = 404
  Caption = 'TeeChart.
TChartShape Component Example'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 153
    Top = 57
    Width = 414
    Height = 313
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    Foot.Frame.Color = clScrollBar
    Title.Frame.Color = clScrollBar
    Title.Text.Strings = (
      'TChart Component')
    BottomAxis.Grid.Color =
clScrollBar
    LeftAxis.Grid.Color = clScrollBar
    Legend.Alignment = laTop
    Legend.Visible = False
    RightAxis.Grid.Color =
clScrollBar
    TopAxis.Grid.Color = clScrollBar
    Align = alClient
    TabOrder = 0
    <-------> object LineSeries1:
TLineSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clRed
      Pointer.InflateMargins = True
      Pointer.Style = psRectangle
      Pointer.Visible = False
      XValues.DateTime = True
      XValues.Name = 'X'
```

```
      Height = 19
      Caption = 'Auto Scale Last Page'
      Font.Charset =
DEFAULT_CHARSET
      Font.Color = clBlack
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 5
      OnClick = CheckBox1Click
    end
    <-------> object BitBtn3: TBitBtn
      Left = 40
      Top = 332
      Width = 93
      Height = 33
      Font.Charset =
DEFAULT_CHARSET
      Font.Color = clBlack
      Font.Height = -11
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 6
      Kind = bkClose
    end
    <-------> object Memo1: TMemo
      Left = 32
      Top = 200
      Width = 121
      Height = 109
      Lines.Strings = (
        'Using the Chart'
        'MaxPointsPerPage'
        'property Series can'
        'be divided in "pages"'
        'so you can navigate'
        'across them.')
      TabOrder = 7
    end
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
<-------> object ShapesForm:
TShapesForm
  Left = 200
```

Marks.Visible = False
SeriesColor = clYellow
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 345
Top = 130
end
<-------> object ChartShape2: TChartShape
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clWhite
Brush.Color = clRed
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'Arial'
Font.Style = []
Pen.Color = clLime
Style = chasLine
X0 = 729090.000000000000000000
X1 = 729108.000000000000000000
Y0 = 692.750000000000000000
Y1 = 2915.500000000000000000
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 161

XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 391
Top = 130
end
<-------> object ChartShape3: TChartShape
Cursor = crCross
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clWhite
OnClick = ChartShape3Click
Brush.Color = clWhite
Brush.Style = bsDiagCross
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'Arial'
Font.Style = []
Style = chasRectangle
X0 = 729090.000000000000000000
X1 = 729108.000000000000000000
Y0 = 549.500000000000000000
Y1 = 1742.000000000000000000
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 161
Top = 206
end
<-------> object LineSeries2: TLineSeries
Marks.ArrowLength = 8

```
    Left = 19
    Top = 4
    Width = 402
    Height = 19
    HelpType = htKeyword
    Caption = 'The Line and Ellipse are TChartShape components.'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -17
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
  end
  <-------> object Label2: TLabel
    Left = 19
    Top = 28
    Width = 426
    Height = 19
    HelpType = htKeyword
    Caption = 'Zoom or Scroll to see how shapes follow Chart Series.'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -17
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
  end
 end
 <-------> object Panel2: TPanel
  Left = 0
  Top = 57
  Width = 153
  Height = 313
  Align = alLeft
  TabOrder = 2
  <-------> object BitBtn2: TBitBtn
    Left = 26
    Top = 256
    Width = 89
    Height = 33
    Caption = 'Close'
    TabOrder = 0
    Kind = bkClose
  end
```

```
    Top = 158
  end
  <-------> object ChartShape1: TChartShape
    Marks.ArrowLength = 8
    Marks.Visible = False
    SeriesColor = clWhite
    Brush.Color = 8454016
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -21
    Font.Name = 'Arial'
    Font.Style = []
    Text.Strings = (
      'Hello'
      'Tee'
      'World !')
    Pen.Color = clNavy
    Pen.Width = 2
    X0 = 729090.000000000000000000
    X1 = 729108.000000000000000000
    Y0 = 737.250000000000000000
    Y1 = 3803.500000000000000000
    XValues.DateTime = False
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
    Left = 157
    Top = 108
  end
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 0
  Width = 567
  Height = 57
  Align = alTop
  TabOrder = 1
  <-------> object Label1: TLabel
```

example of event handling.'
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
BackWall.Color = 8454016
BackWall.Pen.Visible = False
BottomWall.Color = clRed
Foot.Font.Charset = DEFAULT_CHARSET
Foot.Font.Color = clRed
Foot.Font.Height = -15
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsBold]
Foot.Frame.Color = clScrollBar
LeftWall.Color = 16777088
Title.Alignment = taLeftJustify
Title.Brush.Color = clWhite
Title.Brush.Style = bsClear
Title.Color = 16777088
Title.Font.Charset = DEFAULT_CHARSET
Title.Font.Color = clBlue
Title.Font.Height = -16
Title.Font.Name = 'Arial'
Title.Font.Style = [fsBold, fsItalic]
Title.Frame.Color = 8421440
Title.Frame.Width = 4
Title.Text.Strings = (
'Pyramid Bar Chart built from SQL Query.')
OnClickLegend = DBChart1ClickLegend
BackColor = 8454016
BottomAxis.Grid.Color = clScrollBar
Chart3DPercent = 40
ClipPoints = False
Frame.Visible = False
LeftAxis.Grid.Color = clScrollBar
Legend.Color = clAqua
Legend.Font.Charset = DEFAULT_CHARSET
Legend.Font.Color = clBlack
Legend.Font.Height = -16
Legend.Font.Name = 'Arial'
Legend.Font.Style = [fsBold, fsItalic]
Legend.Frame.Color = clTeal
Legend.Frame.Width = 4
Legend.ShadowSize = 11

<-------> object Memo1: TMemo
Left = 16
Top = 44
Width = 121
Height = 161
Lines.Strings = (
'Shapes are simple '
'Series components '
'with special formatting '
'attributes.'
''
'Zoom and scrolling'
'is also possible with'
'Shapes.')
TabOrder = 1
end
end
End
\\\\\\\\\\\\\\\
10
<-------> object SQLBarsForm: TSQLBarsForm
Left = 191
Top = 116
Width = 584
Height = 471
Caption = 'TeeChart SQL Demo'
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
Scaled = False
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> object DBChart1: TDBChart
Left = 0
Top = 85
Width = 576
Height = 181
Hint = 'Click on a Bar to see an

```
  Font.Color = clRed
  Font.Height = -12
  Font.Name = 'Arial'
  Font.Style = [fsItalic]
  ParentFont = False
  TabOrder = 1
  <-------> object BitBtn1: TBitBtn
    Left = 292
    Top = 4
    Width = 89
    Height = 33
    TabOrder = 0
    Kind = bkClose
  end
end
<-------> object Panel2: TPanel
  Left = 0
  Top = 0
  Width = 576
  Height = 85
  Align = alTop
  TabOrder = 2
  <-------> object Label1: TLabel
    Left = 286
    Top = 54
    Width = 45
    Height = 13
    HelpType = htKeyword
    Caption = 'Bar &Style:'
  end
  <-------> object Memo1: TMemo
    Left = 16
    Top = 10
    Width = 261
    Height = 67
    Lines.Strings = (
      'select Name,Weight from ":dbdemos:animals.dbf"'
      'where Weight < 20'
      '')
    TabOrder = 0
  end
  <-------> object BitBtn2: TBitBtn
    Left = 286
    Top = 10
    Width = 111
    Height = 33
    Caption = 'Run Query !!'
```

```
  Legend.TextStyle = ltsRightValue
  Legend.TopPos = 5
  RightAxis.Grid.Color = clScrollBar
  TopAxis.Grid.Color = clScrollBar
  Align = alClient
  ParentShowHint = False
  ShowHint = True
  TabOrder = 0
  <-------> object BarSeries1: TBarSeries
    ColorEachPoint = True
    Cursor = crUpArrow
    Marks.ArrowLength = 20
    Marks.Style = smsPercent
    Marks.Visible = True
    DataSource = Query1
    SeriesColor = clRed
    XLabelsSource = 'NAME'
    OnClick = BarSeries1Click
    BarStyle = bsPyramid
    BarWidthPercent = 80
    XValues.DateTime = False
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
    YValues.ValueSource = 'WEIGHT'
    OnGetBarStyle = BarSeries1GetBarStyle
    Left = 82
    Top = 118
  end
 end
<-------> object Panel1: TPanel
  Left = 0
  Top = 396
  Width = 576
  Height = 41
  Align = alBottom
  Alignment = taLeftJustify
  Font.Charset = DEFAULT_CHARSET
```

```
values !!!'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -16
    Font.Name = 'Arial'
    Font.Style = [fsBold, fsItalic]
    ParentFont = False
  end
  <-------> object DBGrid1:
TDBGrid
    Left = 1
    Top = 1
    Width = 176
    Height = 128
    Align = alLeft
    DataSource = DataSource1
    TabOrder = 0
    TitleFont.Charset =
DEFAULT_CHARSET
    TitleFont.Color = clBlack
    TitleFont.Height = -11
    TitleFont.Name = 'MS Sans
Serif'
    TitleFont.Style = []
  end
 end
 <-------> object DataSource1:
TDataSource
  DataSet = Query1
  Left = 215
  Top = 36
 end
 <-------> object Query1: TQuery
  Active = True
  DatabaseName = 'DBDEMOS'
  RequestLive = True
  SQL.Strings = (
   'select Name,Weight from '
   '":dbdemos:animals.dbf"'
   'where Weight<20'
   '')
  Left = 118
  Top = 34
 end
End
\\\\\\\\\\\\\\\\\\\\
11
```

```
    TabOrder = 1
    OnClick = BitBtn2Click
    Glyph.Data = {
    NumGlyphs = 2
  end
  <-------> object ComboBox1:
TComboBox
    Left = 342
    Top = 50
    Width = 127
    Height = 21
    Style = csDropDownList
    ItemHeight = 13
    TabOrder = 2
    OnChange = ComboBox1Change
    Items.Strings = (
     'Rectangle'
     'Pyramid'
     'InvPyramid'
     'Cilinder'
     'Ellipse'
     'Arrow')
  end
  <-------> object CBRandomBar:
TCheckBox
    Left = 477
    Top = 53
    Width = 88
    Height = 17
    Caption = '<-- &Random'
    TabOrder = 3
    OnClick = CBRandomBarClick
  end
 end
 <-------> object Panel3: TPanel
  Left = 0
  Top = 266
  Width = 576
  Height = 130
  Align = alBottom
  TabOrder = 3
  <-------> object Label2: TLabel
    Left = 184
    Top = 22
    Width = 203
    Height = 19
    HelpType = htKeyword
    Caption = '<--- Try to change
```

LeftAxis.Ticks.Color = clYellow
LeftAxis.Title.Caption = 'Quantity'
Legend.Font.Charset = DEFAULT_CHARSET
Legend.Font.Color = clBlack
Legend.Font.Height = -12
Legend.Font.Name = 'Arial'
Legend.Font.Style = []
Align = alClient
BevelInner = bvRaised
BevelOuter = bvNone
BorderWidth = 14
TabOrder = 0
<-------> object BarSeries1: TBarSeries
Marks.ArrowLength = 20
Marks.Visible = False
SeriesColor = clRed
MultiBar = mbNone
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 108
Top = 4
end
<-------> object BarSeries2: TBarSeries
Marks.ArrowLength = 20
Marks.Visible = False
SeriesColor = clGreen
MultiBar = mbNone
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone

<-------> object StackedForm: TStackedForm
Left = 191
Top = 109
Width = 652
Height = 431
Caption = 'Stacked Bars'
Color = clBtnFace
ParentFont = True
OldCreateOrder = True
Position = poScreenCenter
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> object Chart1: TChart
Left = 0
Top = 41
Width = 510
Height = 356
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
BackWall.Color = 8454016
BottomWall.Color = 16512
Foot.Font.Charset = DEFAULT_CHARSET
Foot.Font.Color = clRed
Foot.Font.Height = -12
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsItalic]
Foot.Frame.Color = clScrollBar
Gradient.Direction = gdRightLeft
Gradient.StartColor = 16744576
LeftWall.Color = clBlue
Title.Frame.Color = clScrollBar
Title.Text.Strings = (
'TChart Component')
BackColor = 8454016
BottomAxis.DateTimeFormat = 'M/d/yy'
BottomAxis.Title.Caption = 'Sales Date'
Chart3DPercent = 60
LeftAxis.MinorTickCount = 1
LeftAxis.MinorTickLength = 5
LeftAxis.MinorTicks.Color = clBlue
LeftAxis.TickLength = 8

```
  <-------> object Label1: TLabel
    Left = 7
    Top = 5
    Width = 32
    Height = 13
    HelpType = htKeyword
    Caption = '&Series:'
    FocusControl = ComboBox1
  end
  <-------> object Shape1: TShape
    Left = 7
    Top = 52
    Width = 118
    Height = 29
    HelpType = htKeyword
    OnMouseUp = Shape1MouseUp
  end
  <-------> object Button1: TButton
    Left = 17
    Top = 86
    Width = 99
    Height = 29
    Caption = 'S&croll Series'
    TabOrder = 0
    OnClick = Button1Click
  end
  <-------> object RadioGroup2:
TRadioGroup
    Left = 13
    Top = 150
    Width = 108
    Height = 139
    Caption = '&Bar Style:'
    ItemIndex = 0
    Items.Strings = (
      'Rectangle'
      'Pyramid'
      'InvPyramid'
      'Cilinder'
      'Ellipse'
      'Arrow'
      'Gradient')
    TabOrder = 1
    OnClick = RadioGroup2Click
  end
  <-------> object ComboBox1:
TComboBox
    Left = 9
```

```
      Left = 141
      Top = 4
    end
    <-------> object BarSeries3:
TBarSeries
      Marks.ArrowLength = 20
      Marks.Visible = False
      SeriesColor = clAqua
      MultiBar = mbNone
      XValues.DateTime = True
      XValues.Name = 'X'
      XValues.Multiplier =
1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier =
1.000000000000000000
      YValues.Order = loNone
      Left = 172
      Top = 4
    end
  end
  <-------> object RadioGroup1:
TRadioGroup
    Left = 0
    Top = 0
    Width = 644
    Height = 41
    Align = alTop
    Caption = 'Bar &Position:'
    Columns = 4
    ItemIndex = 0
    Items.Strings = (
      'Behind'
      'Side to Side'
      'Stacked'
      'Stacked 100%')
    TabOrder = 1
    OnClick = RadioGroup1Click
  end
  <-------> object Panel1: TPanel
    Left = 510
    Top = 41
    Width = 134
    Height = 356
    Align = alRight
    TabOrder = 2
```

```
    Interval = 1
    OnTimer = Timer1Timer
    Left = 452
    Top = 18
  end
End

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
12

<-------> object TablePieForm: TTablePieForm
  Left = 193
  Top = 109
  Width = 629
  Height = 440
  Caption = 'TeeChart Table-Pie Demo'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -9
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object DBChart1: TDBChart
    Left = 0
    Top = 0
    Width = 621
    Height = 255
    AllowPanning = pmNone
    AllowZoom = False
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    Foot.Font.Charset = DEFAULT_CHARSET
    Foot.Font.Color = clRed
    Foot.Font.Height = -15
    Foot.Font.Name = 'Arial'
    Foot.Font.Style = [fsBold]
    Foot.Frame.Color = clScrollBar
    Foot.Text.Strings = (
    Top = 23
    Width = 116
    Height = 21
    Style = csDropDownList
    ItemHeight = 13
    TabOrder = 2
    OnChange = ComboBox1Change
  end
  <-------> object CheckBox1: TCheckBox
    Left = 9
    Top = 123
    Width = 120
    Height = 19
    Caption = '&Animate !!!'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clRed
    Font.Height = -19
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 3
    OnClick = CheckBox1Click
  end
  <-------> object BitBtn3: TBitBtn
    Left = 25
    Top = 320
    Width = 89
    Height = 33
    TabOrder = 4
    Kind = bkClose
  end
  <-------> object CheckBox2: TCheckBox
    Left = 43
    Top = 296
    Width = 50
    Height = 17
    Caption = '3D'
    Checked = True
    State = cbChecked
    TabOrder = 5
    OnClick = CheckBox2Click
  end
 end
 <-------> object Timer1: TTimer
  Enabled = False
```

Marks.BackColor = clWhite
Marks.Font.Charset = DEFAULT_CHARSET
Marks.Font.Color = clRed
Marks.Font.Height = -13
Marks.Font.Name = 'Arial'
Marks.Font.Style = [fsItalic]
Marks.Frame.Color = clNavy
Marks.Frame.Width = 3
Marks.Style = smsLabelValue
Marks.Visible = True
DataSource = Table1
SeriesColor = clRed
XLabelsSource = 'NAME'
OnClick = PieSeries1Click
Circled = True
OtherSlice.Text = 'Other'
PieValues.DateTime = False
PieValues.Name = 'Y'
PieValues.Multiplier = 1.000000000000000000
PieValues.Order = loNone
PieValues.ValueSource = 'SIZE'
Left = 187
Top = 6
end
end
<-------> object Panel2: TPanel
Left = 0
Top = 255
Width = 621
Height = 117
Align = alBottom
TabOrder = 1
<-------> object DBGrid1: TDBGrid
Left = 349
Top = 1
Width = 271
Height = 115
Align = alRight
DataSource = DataSource1
TabOrder = 0
TitleFont.Charset = DEFAULT_CHARSET
TitleFont.Color = clBlack
TitleFont.Height = -9
TitleFont.Name = 'MS Sans

'XLabelSource:=Table1Name'
'PieValueSource:=Table1Size')
Title.Font.Charset = DEFAULT_CHARSET
Title.Font.Color = clBlue
Title.Font.Height = -16
Title.Font.Name = 'Arial'
Title.Font.Style = [fsBold, fsItalic]
Title.Frame.Color = clScrollBar
Title.Text.Strings = (
'Pie Chart built from Table :DBDEMOS:ANIMALS.DBF')
AxisVisible = False
BottomAxis.Grid.Color = clScrollBar
ClipPoints = False
Frame.Visible = False
LeftAxis.Grid.Color = clScrollBar
Legend.Alignment = laLeft
Legend.Color = clAqua
Legend.Font.Charset = DEFAULT_CHARSET
Legend.Font.Color = clBlack
Legend.Font.Height = -16
Legend.Font.Name = 'Arial'
Legend.Font.Style = [fsBold, fsItalic]
Legend.Frame.Color = clTeal
Legend.Frame.Width = 4
Legend.ShadowColor = clGray
Legend.ShadowSize = 11
Legend.TextStyle = ltsLeftPercent
RightAxis.Grid.Color = clScrollBar
TopAxis.Grid.Color = clScrollBar
View3DOptions.Elevation = 315
View3DOptions.Orthogonal = False
View3DOptions.Perspective = 0
View3DOptions.Rotation = 360
View3DWalls = False
Align = alClient
TabOrder = 0
<-------> object PieSeries1: TPieSeries
Cursor = crDrag
Marks.Arrow.Color = clYellow
Marks.Arrow.Width = 2
Marks.ArrowLength = 11

```
    Left = 0
    Top = 372
    Width = 621
    Height = 34
    Align = alBottom
    Alignment = taLeftJustify
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clRed
    Font.Height = -9
    Font.Name = 'Arial'
    Font.Style = [fsItalic]
    ParentFont = False
    TabOrder = 2
  end
  <-------> object Table1: TTable
    Active = True
    DatabaseName = 'DBDEMOS'
    TableName = 'ANIMALS.DBF'
    Left = 29
    Top = 344
    <-------> object Table1NAME:
TStringField
      FieldName = 'NAME'
      Size = 10
    end
    <-------> object Table1SIZE:
TSmallintField
      FieldName = 'SIZE'
    end
    <-------> object Table1WEIGHT:
TSmallintField
      FieldName = 'WEIGHT'
    end
  end
  <-------> object DataSource1:
TDataSource
    DataSet = Table1
    Left = 111
    Top = 344
  end
End
\\\\\\\\\\\\\\\\\\\\\\\
13
<-------> object TeeMainForm:
TTeeMainForm
  Left = 192
  Top = 109
```

```
Serif'
      TitleFont.Style = []
    end
    <-------> object CheckBox1:
TCheckBox
      Left = 10
      Top = 8
      Width = 101
      Height = 14
      Caption = 'ACTIVE Table'
      Checked = True
      State = cbChecked
      TabOrder = 1
      OnClick = CheckBox1Click
    end
    <-------> object RadioGroup1:
TRadioGroup
      Left = 10
      Top = 32
      Width = 131
      Height = 32
      Caption = 'Use Field:'
      Columns = 2
      ItemIndex = 0
      Items.Strings = (
        'Size'
        'Weight')
      TabOrder = 2
      OnClick = RadioGroup1Click
    end
    <-------> object BitBtn1: TBitBtn
      Left = 27
      Top = 79
      Width = 90
      Height = 27
      Caption = 'Close'
      Font.Charset =
DEFAULT_CHARSET
      Font.Color = clBlack
      Font.Height = -11
      Font.Name = 'Arial'
      Font.Style = []
      ParentFont = False
      TabOrder = 3
      Kind = bkClose
    end
  end
  <-------> object Panel1: TPanel
```

```
<-------> object Panel1: TPanel
  Left = 0
  Top = 376
  Width = 632
  Height = 30
  Align = alBottom
  Alignment = taLeftJustify
  TabOrder = 0
  <-------> object Label11: TLabel
    Left = 12
    Top = 49
    Width = 3
    Height = 13
    HelpType = htKeyword
  end
  <-------> object Label13: TLabel
    Left = 12
    Top = 6
    Width = 129
    Height = 19
    HelpType = htKeyword
    Caption = 'Steema Software'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -16
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
  end
  <-------> object Label15: TLabel
    Left = 212
    Top = 9
    Width = 27
    Height = 13
    HelpType = htKeyword
    Caption = 'email:'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
  end
  <-------> object Label16: TLabel
    Left = 244
    Top = 9
```

```
  ActiveControl = Button2
  BorderIcons = [biSystemMenu]
  BorderStyle = bsDialog
  Caption = 'TeeChart Standard
version 4.0 - Demo'
  ClientHeight = 406
  ClientWidth = 632
  Color = clNavy
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Image1: TImage
    Left = 484
    Top = 26
    Width = 141
    Height = 45
    HelpType = htKeyword
    AutoSize = True
    Picture.Data = {
    OnClick = Label10Click
  end
  <-------> object Label2: TLabel
    Left = 168
    Top = 353
    Width = 300
    Height = 13
    HelpType = htKeyword
    Caption = '© Copyright 1996-2001
by David Berneda. All Rights
Reserved.'
    Color = clNavy
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clWhite
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentColor = False
    ParentFont = False
  end
```

```
  TabOrder = 1
  <-------> object Button2: TButton
    Left = 8
    Top = -1
    Width = 141
    Height = 41
    Caption = 'Show me the Demo !'
    Default = True
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -12
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 0
    OnClick = Button2Click
  end
  <-------> object Button3: TButton
    Left = 8
    Top = 85
    Width = 141
    Height = 29
    Caption = 'Features...'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -12
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 1
    OnClick = Button3Click
  end
  <-------> object Button4: TButton
    Left = 8
    Top = 122
    Width = 141
    Height = 29
    Caption = 'Pro Specifications...'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -12
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 2
    Width = 85
    Height = 13
    HelpType = htKeyword
    Caption = 'info@steema.com'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
  end
  <-------> object Label3: TLabel
    Left = 375
    Top = 6
    Width = 119
    Height = 16
    Cursor = crHandPoint
    HelpType = htKeyword
    Caption = 'www.steema.com'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlue
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold, fsUnderline]
    ParentFont = False
    OnClick = Label3Click
  end
  <-------> object Button1: TButton
    Left = 536
    Top = 4
    Width = 89
    Height = 23
    Caption = 'Close'
    ModalResult = 1
    TabOrder = 0
    OnClick = Button1Click
  end
 end
 <-------> object TPanel
  Left = 476
  Top = 102
  Width = 157
  Height = 251
  BevelOuter = bvNone
  BorderWidth = 1
  Color = clNavy
```

```
      Font.Charset =
DEFAULT_CHARSET
      Font.Color = clRed
      Font.Height = -29
      Font.Name = 'Times New
Roman'
      Font.Style = [fsBold]
      ParentFont = False
      Transparent = True
      OnClick = Label10Click
      OnMouseDown =
Chart2MouseDown
    end
    <-------> object FastLineSeries1:
TFastLineSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clAqua
      LinePen.Color = clAqua
      XValues.DateTime = False
      XValues.Name = 'X'
      XValues.Multiplier =
1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier =
1.000000000000000000
      YValues.Order = loNone
    end
    <-------> object FastLineSeries2:
TFastLineSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clYellow
      LinePen.Color = clYellow
      XValues.DateTime = True
      XValues.Name = 'X'
      XValues.Multiplier =
1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier =
1.000000000000000000
      YValues.Order = loNone
    end
  end
end
```

```
    OnClick = Button4Click
  end
  <-------> object Button6: TButton
    Left = 8
    Top = 48
    Width = 141
    Height = 29
    Caption = 'Chart Styles...'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -12
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 3
    OnClick = Button6Click
  end
  <-------> object Chart2: TChart
    Left = 8
    Top = 168
    Width = 141
    Height = 73
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    Title.Text.Strings = (
      'TChart')
    Title.Visible = False
    AxisVisible = False
    ClipPoints = False
    Frame.Visible = False
    Legend.Color = clSilver
    Legend.Visible = False
    View3DWalls = False
    BevelWidth = 2
    Color = clSilver
    TabOrder = 4
    OnMouseDown =
Chart2MouseDown
    OnMouseUp = Chart2MouseUp
    <-------> object Label12: TLabel
      Left = 25
      Top = 16
      Width = 97
      Height = 32
      HelpType = htKeyword
      Caption = 'About...'
```

```
      Marks.Visible = False
      SeriesColor = clRed
      CustomYRadius = 25
      ExplodeBiggest = 25
      OtherSlice.Text = 'Other'
      PieValues.DateTime = False
      PieValues.Name = 'Pie'
      PieValues.Multiplier =
1.000000000000000000
      PieValues.Order = loNone
    end
  end
  <-------> object Chart8: TChart
    Left = 3
    Top = 163
    Width = 97
    Height = 66
    AllowPanning = pmNone
    AllowZoom = False
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    MarginBottom = 10
    MarginLeft = 5
    MarginRight = 5
    MarginTop = 0
    Title.Font.Charset =
DEFAULT_CHARSET
    Title.Font.Color = clBlack
    Title.Font.Height = -17
    Title.Font.Name = 'Arial'
    Title.Font.Style = [fsBold]
    Title.Text.Strings = (
      '')
    AxisVisible = False
    Chart3DPercent = 95
    ClipPoints = False
    Frame.Visible = False
    Legend.Visible = False
    View3DOptions.Elevation = 315
    View3DOptions.Orthogonal =
False
    View3DOptions.Perspective = 0
    View3DOptions.Rotation = 360
    View3DOptions.Zoom = 87
    View3DWalls = False
    BevelOuter = bvNone
    Color = clAqua
```

```
  <-------> object Panel2: TPanel
    Left = 9
    Top = 102
    Width = 465
    Height = 241
    BevelWidth = 2
    Color = clAqua
    TabOrder = 2
    <-------> object Chart7: TChart
      Left = 3
      Top = 7
      Width = 97
      Height = 66
      AllowPanning = pmNone
      AllowZoom = False
      BackWall.Brush.Color = clWhite
      BackWall.Brush.Style = bsClear
      BackWall.Pen.Visible = False
      MarginBottom = 10
      MarginLeft = 5
      MarginRight = 5
      MarginTop = 0
      Title.Font.Charset =
DEFAULT_CHARSET
      Title.Font.Color = clBlack
      Title.Font.Height = -17
      Title.Font.Name = 'Arial'
      Title.Font.Style = [fsBold]
      Title.Text.Strings = (
        '')
      AxisVisible = False
      Chart3DPercent = 50
      ClipPoints = False
      Frame.Visible = False
      Legend.Visible = False
      View3DOptions.Elevation = 315
      View3DOptions.Orthogonal =
False
      View3DOptions.Perspective = 0
      View3DOptions.Rotation = 360
      View3DOptions.Zoom = 92
      View3DWalls = False
      BevelOuter = bvNone
      Color = clAqua
      TabOrder = 1
      <-------> object PieSeries6:
TPieSeries
        Marks.ArrowLength = 8
```

```
    View3DOptions.Zoom = 92
    View3DWalls = False
    BevelOuter = bvNone
    Color = clAqua
    TabOrder = 4
    <-------> object PieSeries9:
TPieSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clRed
      CustomYRadius = 25
      ExplodeBiggest = 40
      OtherSlice.Text = 'Other'
      PieValues.DateTime = False
      PieValues.Name = 'Pie'
      PieValues.Multiplier =
1.000000000000000000
      PieValues.Order = loNone
    end
  end
  <-------> object Chart9: TChart
    Left = 206
    Top = 4
    Width = 255
    Height = 221
    AllowPanning = pmNone
    AllowZoom = False
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    MarginBottom = 10
    MarginLeft = 5
    MarginRight = 5
    MarginTop = 0
    Title.Font.Charset =
DEFAULT_CHARSET
    Title.Font.Color = clBlack
    Title.Font.Height = -24
    Title.Font.Name = 'Arial'
    Title.Font.Style = [fsBold]
    Title.Text.Strings = (
      '')
    Title.Visible = False
    AxisVisible = False
    Chart3DPercent = 70
    ClipPoints = False
    Frame.Visible = False
    Legend.TextStyle = ltsPlain
```

```
    TabOrder = 2
    <-------> object PieSeries7:
TPieSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clRed
      CustomYRadius = 25
      ExplodeBiggest = 20
      OtherSlice.Text = 'Other'
      PieValues.DateTime = False
      PieValues.Name = 'Pie'
      PieValues.Multiplier =
1.000000000000000000
      PieValues.Order = loNone
    end
  end
  <-------> object Chart10: TChart
    Left = 105
    Top = 139
    Width = 97
    Height = 66
    AllowPanning = pmNone
    AllowZoom = False
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    MarginBottom = 10
    MarginLeft = 5
    MarginRight = 5
    MarginTop = 0
    Title.Font.Charset =
DEFAULT_CHARSET
    Title.Font.Color = clBlack
    Title.Font.Height = -17
    Title.Font.Name = 'Arial'
    Title.Font.Style = [fsBold]
    Title.Text.Strings = (
      '')
    AxisVisible = False
    Chart3DPercent = 100
    ClipPoints = False
    Frame.Visible = False
    Legend.Visible = False
    View3DOptions.Elevation = 315
    View3DOptions.Orthogonal =
False
    View3DOptions.Perspective = 0
    View3DOptions.Rotation = 360
```

```
    Top = 31
    Width = 97
    Height = 66
    AllowPanning = pmNone
    AllowZoom = False
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    MarginBottom = 10
    MarginLeft = 5
    MarginRight = 5
    MarginTop = 0
    Title.Font.Charset =
DEFAULT_CHARSET
    Title.Font.Color = clBlack
    Title.Font.Height = -17
    Title.Font.Name = 'Arial'
    Title.Font.Style = [fsBold]
    Title.Text.Strings = (
      '')
    AxisVisible = False
    Chart3DPercent = 50
    ClipPoints = False
    Frame.Visible = False
    Legend.Visible = False
    View3DOptions.Elevation = 315
    View3DOptions.Orthogonal =
False
    View3DOptions.Perspective = 0
    View3DOptions.Rotation = 360
    View3DWalls = False
    BevelOuter = bvNone
    Color = clAqua
    TabOrder = 0
    <-------> object Series1:
TPieSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clRed
      CustomYRadius = 25
      ExplodeBiggest = 25
      OtherSlice.Text = 'Other'
      PieValues.DateTime = False
      PieValues.Name = 'Pie'
      PieValues.Multiplier =
1.000000000000000000
      PieValues.Order = loNone
    end
```

```
    Legend.Visible = False
    View3DOptions.Elevation = 315
    View3DOptions.Orthogonal =
False
    View3DOptions.Perspective = 0
    View3DOptions.Rotation = 360
    View3DWalls = False
    BevelOuter = bvNone
    Color = clAqua
    TabOrder = 3
    <-------> object PieSeries8:
TPieSeries
      Marks.Arrow.Color = clBlack
      Marks.ArrowLength = 1
      Marks.Font.Charset =
DEFAULT_CHARSET
      Marks.Font.Color = clNavy
      Marks.Font.Height = -11
      Marks.Font.Name = 'Arial'
      Marks.Font.Style = []
      Marks.Frame.Visible = False
      Marks.Transparent = True
      Marks.Visible = True
      DataSource = Series1
      PercentFormat = '##0.00 %'
      SeriesColor = clRed
      CustomYRadius = 35
      ExplodeBiggest = 40
      OtherSlice.Text = 'Other'
      PiePen.Visible = False
      PieValues.DateTime = False
      PieValues.Name = 'Pie'
      PieValues.Multiplier =
1.000000000000000000
      PieValues.Order = loNone
      DataSources = (
        'Series1'
        'PieSeries6'
        'PieSeries7'
        'PieSeries9')
      <-------> object
PieSeries8TAddTeeFunction:
TAddTeeFunction
      end
    end
  end
  <-------> object Chart1: TChart
    Left = 105
```

```
      Caption = 'For Borland Delphi, Kylix and C++ Builder'
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clNavy
      Font.Height = -16
      Font.Name = 'Arial'
      Font.Style = [fsBold]
      ParentFont = False
    end
  end
  <-------> object Timer1: TTimer
    Interval = 1
    OnTimer = Timer1Timer
    Left = 408
    Top = 283
  end
  <-------> object Timer2: TTimer
    Interval = 200
    OnTimer = Timer2Timer
    Left = 356
    Top = 281
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
14
<-------> object AxisLabelsForm: TAxisLabelsForm
  Left = 197
  Top = 109
  Width = 501
  Height = 355
  Caption = 'TeeChart Sample Project.  Custom Axis Labeling'
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 153
    Top = 0
    Width = 340
    Height = 321
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    end
  end
  <-------> object TPanel
    Left = 8
    Top = 359
    Width = 156
    Height = 1
    TabOrder = 3
  end
  <-------> object TPanel
    Left = 475
    Top = 360
    Width = 153
    Height = 1
    TabOrder = 4
  end
  <-------> object Panel3: TPanel
    Left = 8
    Top = 11
    Width = 465
    Height = 81
    TabOrder = 5
    <-------> object Label1: TLabel
      Left = 8
      Top = 15
      Width = 449
      Height = 22
      HelpType = htKeyword
      Alignment = taCenter
      AutoSize = False
      Caption = 'The 100% Native Charting VCL / CLX Library'
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -19
      Font.Name = 'Arial'
      Font.Style = [fsBold]
      ParentFont = False
    end
    <-------> object Label9: TLabel
      Left = 8
      Top = 44
      Width = 449
      Height = 19
      HelpType = htKeyword
      Alignment = taCenter
      AutoSize = False
```

<-------> object PointSeries1: TPointSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clGreen
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = True
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 283
Top = 132
end
end
<-------> object Panel1: TPanel
Left = 0
Top = 0
Width = 153
Height = 321
Align = alLeft
TabOrder = 1
<-------> object RadioGroup1: TRadioGroup
Left = 17
Top = 16
Width = 120
Height = 57
Caption = 'Choose Axis Labels:'
ItemIndex = 1
Items.Strings = (
'Default'
'User defined')
TabOrder = 0
OnClick = RadioGroup1Click
end
<-------> object BitBtn3: TBitBtn
Left = 32
Top = 328
Width = 89
Height = 33

Foot.Frame.Color = clScrollBar
Title.Frame.Color = clScrollBar
Title.Text.Strings = (
'TChart Component')
BottomAxis.LabelsFont.Charset = DEFAULT_CHARSET
BottomAxis.LabelsFont.Color = clMaroon
BottomAxis.LabelsFont.Height = -21
BottomAxis.LabelsFont.Name = 'Arial'
BottomAxis.LabelsFont.Style = [fsBold, fsItalic]
LeftAxis.LabelsFont.Charset = DEFAULT_CHARSET
LeftAxis.LabelsFont.Color = clNavy
LeftAxis.LabelsFont.Height = -24
LeftAxis.LabelsFont.Name = 'Arial'
LeftAxis.LabelsFont.Style = [fsBold, fsItalic]
Legend.Alignment = laTop
OnGetNextAxisLabel = Chart1GetNextAxisLabel
Align = alClient
TabOrder = 0
<-------> object LineSeries1: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clRed
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 199
Top = 122
end

<-------> object Memo1: TMemo
Left = 0
Top = 167
Width = 467
Height = 307
Align = alClient
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
Lines.Strings = (
'Some of the TeeChart Pro 5.0 features are:'
''
'** Supports Kylix.'
'** Optional FULL 100% source code and Pro-Support at additional ' +
'price'
'** Electronic delivery. Optional CDRom and printed guide mailed.'
'** Multiple Charts per Form and Multiple Series per Chart.'
'** New ! Multiple unlimited axes per Chart.'
'** New ! 3D Rotation, elevation, zoom and scroll.'
'** New ! Optional OpenGL 3D realistic rendering'
'** New ! Multi-row Legends'
'** New ! Logarithmic axis labels'
'** New ! Radar, Surface 3D, Contour 3D, Point 3D, and Error-Seri' +
'es'
'** New ! Pie Slice Exploding'
'** New ! Custom Arrays supported.'
'** Up to 2500000000 points per Series !'
'** Add, change, delete values manually or...'
'** Link directly to Tables and Queries. (TDBChart)'

TabOrder = 1
Kind = bkClose
end
<-------> object Memo1: TMemo
Left = 16
Top = 96
Width = 121
Height = 193
Lines.Strings = (
'This example shows '
'how Axis labels can be '
'customized, '
'both their position and '
'text using '
'OnGetAxisLabel and'
'OnGetNextAxisLabel '
'events.'
''
'In this example Bottom '
'axis labels are set to the '
'starting day of each'
'month.')
TabOrder = 2
end
end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\
15
<-------> object ChartSpecs: TChartSpecs
Left = 192
Top = 106
Width = 475
Height = 508
ActiveControl = Memo1
Caption = 'TeeChart Pro 5.0 Specifications...'
Color = clBtnFace
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
PixelsPerInch = 96
TextHeight = 13

Stacked B' +
'ars...'
' Stacked 100%, Stack Groups'**
' Error Bars, Arrows'**
' Bubble, Pie, Gantt, Candle-Stick, Curve Fitting...'**
' Waterfall, ColorGrid, Histogram, Smith'**
' Funnel, Bollinger, ADX, MACD, Pyramid, Donut, Box-Whisker'**
' Automatic paging of Series points'**
' Custom drawing and printing'**
' Gradient and Picture Background filling'**
' Many new samples and example projects'**
' PacMan game made with TeeChart.'**
' No Royalties or Runtime fees.'**
''
'(*) QuickReport is copyright by Allan Lochert, QuSoft AS, Norway' +
'.')
ParentFont = False
ReadOnly = True
ScrollBars = ssBoth
TabOrder = 0
end
<-------> object Panel2: TPanel
Left = 0
Top = 0
Width = 467
Height = 60
Align = alTop
TabOrder = 1
<-------> object Label1: TLabel
Left = 164
Top = 35
Width = 160
Height = 16
HelpType = htKeyword
Caption = 'Product Specifications.'

' Any Series can be linked to another Series (both at Design an' +**
'd RunTime).'
' Non Data-Aware charting if desired. '**
' Seamless integration with QuickReport 2.0 and 3.0'**
' Automatic Resize and Refreshing'**
' Animated Zoom, Scrolling & RealTime capabilities.'**
' 2D 3D. Z-Order configurable.'**
' Standard, Statistical & Financial Series types included.'**
' Extended Series types and Functions are easy to create.'**
' Tones of properties to customize your preferences.'**
' Events to control user''s input and override default behaviour' +**
's'
' Resource messages (*.rc) to allow internationalization'**
' Property Editor dialogs also available at RunTime.'**
' Complete Printing control both to Printer and User Canvas.'**
' Remote loading of charts from www URL''s'**
' JPEG, PNG, GIF, PCX, Metafile *.WMF and Bitmap *.BMP file and' +**
' clipboard support'
' Accepts any child control (TeeChart derives from TPanel).'**
' No VBX''s or OCX''s required. Just packages and units !'**
' No drawing flickering.'**
' Mini-charts capable'**
' Many Example Projects and a Tips Database'**
' Lines, Scatter, Area, Vertical and Horizontal Bars,**

```
more than 250 new features over the '
      'standard version included in Delphi and C++ Builder.'
      ''

        'Visit us at http://www.steema.com and download the evaluation ve' +
        'rsion.'
      '')
    TabOrder = 0
  end
 end
End
\\\\\\\\\\\\\\\\\\\\
16
<-------> object ColoredForm: TColoredForm
 Left = 189
 Top = 111
 Width = 574
 Height = 439
 Caption = 'TeeChart Custom Coloring Example'
 Color = clBtnFace
 ParentFont = True
 OldCreateOrder = True
 Position = poScreenCenter
 OnCreate = FormCreate
 PixelsPerInch = 96
 TextHeight = 13
 <-------> object Chart1: TChart
  Left = 137
  Top = 0
  Width = 429
  Height = 405
  BackWall.Brush.Color = clWhite
  BackWall.Brush.Style = bsClear
  Foot.Font.Charset = DEFAULT_CHARSET
  Foot.Font.Color = clRed
  Foot.Font.Height = -12
  Foot.Font.Name = 'Arial'
  Foot.Font.Style = [fsItalic]
  Title.Alignment = taRightJustify
  Title.Text.Strings = (
    'Each Point has it''s own Color')
  Chart3DPercent = 10
```

```
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentFont = False
  end
  <-------> object Image2: TImage
    Left = 14
    Top = 8
    Width = 141
    Height = 45
    HelpType = htKeyword
    AutoSize = True
    Picture.Data = {
  end
  <-------> object BitBtn1: TBitBtn
    Left = 368
    Top = 20
    Width = 89
    Height = 33
    TabOrder = 0
    Kind = bkClose
  end
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 60
  Width = 467
  Height = 107
  Align = alTop
  TabOrder = 2
  <-------> object Memo2: TMemo
    Left = 7
    Top = 5
    Width = 450
    Height = 92
    Lines.Strings = (

        'TeeChart Pro 5.0 is a library of 100% Native Borland Delphi char' +
        'ting '
      'and graphing components ready to include in your applications.'
      'TeeChart Pro 5.0 includes
```

```
  Height = 405
  Align = alLeft
  TabOrder = 1
  <-------> object CheckBox1: TCheckBox
    Left = 7
    Top = 20
    Width = 114
    Height = 17
    Caption = 'Left Axis Inverted'
    Checked = True
    State = cbChecked
    TabOrder = 0
    OnClick = CheckBox1Click
  end
  <-------> object BitBtn2: TBitBtn
    Left = 19
    Top = 316
    Width = 89
    Height = 33
    TabOrder = 1
    Kind = bkClose
  end
  <-------> object Memo1: TMemo
    Left = 8
    Top = 72
    Width = 121
    Height = 161
    Lines.Strings = (
      'This demo shows how '
      'to set a different color '
      'for each point after'
      'points have been added '
      'to the Series.'
      ''
      'It shows too how to '
      'invert Axis using the axis'
      'Inverted property.')
    TabOrder = 2
  end
 end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
17
<-------> object CustomAxisForm: TCustomAxisForm
 Left = 166
 Top = 110
```

```
  LeftAxis.Inverted = True
  Legend.Alignment = laBottom
  Align = alClient
  TabOrder = 0
  <-------> object LineSeries1: TLineSeries
    Marks.ArrowLength = 8
    Marks.Visible = False
    SeriesColor = clRed
    Pointer.InflateMargins = True
    Pointer.Style = psRectangle
    Pointer.Visible = False
    Stairs = True
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
  end
  <-------> object PointSeries1: TPointSeries
    Marks.ArrowLength = 8
    Marks.Visible = False
    SeriesColor = clGreen
    Pointer.InflateMargins = True
    Pointer.Style = psRectangle
    Pointer.Visible = True
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
  end
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 0
  Width = 137
```

```
    Marks.ArrowLength = 8
    Marks.Visible = False
    SeriesColor = clRed
    AfterDrawValues = LineSeries1AfterDrawValues
    Pointer.InflateMargins = True
    Pointer.Style = psRectangle
    Pointer.Visible = False
    XValues.DateTime = False
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
    Left = 101
    Top = 63
  end
  <-------> object PointSeries1: TPointSeries
    ColorEachPoint = True
    Marks.ArrowLength = 8
    Marks.Visible = False
    SeriesColor = clGreen
    Pointer.InflateMargins = True
    Pointer.Style = psDiamond
    Pointer.Visible = True
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
    Left = 147
    Top = 61
  end
  <-------> object FastLineSeries1: TFastLineSeries
    Marks.ArrowLength = 8
    Marks.Visible = False
    SeriesColor = clBlue
    LinePen.Color = clBlue
```

```
  Width = 603
  Height = 402
  Caption = 'TeeChart Custom Axis Drawing Example'
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 137
    Top = 0
    Width = 458
    Height = 368
    AnimatedZoom = True
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    Foot.Font.Charset = DEFAULT_CHARSET
    Foot.Font.Color = clRed
    Foot.Font.Height = -12
    Foot.Font.Name = 'Arial'
    Foot.Font.Style = [fsItalic]
    Foot.Frame.Color = clScrollBar
    Gradient.Visible = True
    Title.Frame.Color = clScrollBar
    Title.Text.Strings = (
      'TChart Component')
    Title.Visible = False
    BottomAxis.Grid.Color = clScrollBar
    BottomAxis.Grid.Visible = False
    LeftAxis.Grid.Color = clScrollBar
    LeftAxis.Grid.Visible = False
    Legend.Alignment = laLeft
    RightAxis.Grid.Color = clScrollBar
    RightAxis.Grid.Visible = False
    TopAxis.Grid.Color = clScrollBar
    TopAxis.Grid.Visible = False
    View3D = False
    View3DWalls = False
    Align = alClient
    TabOrder = 0
    <-------> object LineSeries1: TLineSeries
```

```
    Left = 9
    Top = 92
    Width = 97
    Height = 17
    Caption = 'Invert Axis '
    TabOrder = 2
    OnClick = CheckBox2Click
  end
  <-------> object DrawGrid: TCheckBox
    Left = 9
    Top = 60
    Width = 80
    Height = 17
    Caption = 'Draw Grid'
    Checked = True
    State = cbChecked
    TabOrder = 3
    OnClick = DrawGridClick
  end
  <-------> object Memo1: TMemo
    Left = 8
    Top = 136
    Width = 121
    Height = 153
    Lines.Strings = (
      'This example shows '
      'how to draw custom '
      'Axis at specific '
      'positions.'
      ''
      'The Axis CustomDraw '
      'method is use to display '
      'custom Axis.')
    TabOrder = 4
  end
 end
 <-------> object Timer1: TTimer
  Enabled = False
  Interval = 1
  OnTimer = Timer1Timer
  Left = 31
  Top = 34
 end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
18
<-------> object CrossHairForm:
```

```
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
    Left = 187
    Top = 64
  end
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 0
  Width = 137
  Height = 368
  Align = alLeft
  TabOrder = 1
  <-------> object CheckBox1: TCheckBox
    Left = 9
    Top = 12
    Width = 97
    Height = 17
    Caption = '&Animate !!!'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -16
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 0
    OnClick = CheckBox1Click
  end
  <-------> object BitBtn1: TBitBtn
    Left = 25
    Top = 324
    Width = 80
    Height = 33
    TabOrder = 1
    Kind = bkClose
  end
  <-------> object CheckBox2: TCheckBox
```

Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 199
Top = 121
end
end
<-------> object Panel1: TPanel
Left = 0
Top = 0
Width = 145
Height = 370
Align = alLeft
TabOrder = 1
<-------> object Label1: TLabel
Left = 44
Top = 12
Width = 15
Height = 13
HelpType = htKeyword
Caption = '0,0'
end
<-------> object Label2: TLabel
Left = 16
Top = 12
Width = 20
Height = 13
HelpType = htKeyword
Caption = 'X,Y:'
end
<-------> object BitBtn1: TBitBtn
Left = 26
Top = 324
Width = 89
Height = 33
TabOrder = 0
Kind = bkClose
end
<-------> object CheckBox1:

TCrossHairForm
Left = 200
Top = 111
Width = 506
Height = 404
ActiveControl = Chart1
Caption = 'TeeChart Cross Hair Demo'
Color = clBtnFace
ParentFont = True
OldCreateOrder = True
Position = poScreenCenter
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> object Chart1: TChart
Left = 145
Top = 0
Width = 353
Height = 370
Cursor = crCross
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
Foot.Font.Charset = DEFAULT_CHARSET
Foot.Font.Color = clRed
Foot.Font.Height = -12
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsItalic]
Title.Text.Strings = (
'TChart Component')
Title.Visible = False
BottomAxis.Grid.Visible = False
Chart3DPercent = 30
LeftAxis.Grid.Visible = False
View3D = False
Align = alClient
TabOrder = 0
OnMouseMove = Chart1MouseMove
<-------> object LineSeries1: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clRed
AfterDrawValues = LineSeries1AfterDrawValues
Pointer.InflateMargins = True

Caption = 'TeeChart Example. Horizontal and Vertical Bars with DBChart'
Color = clBtnFace
ParentFont = True
OldCreateOrder = True
Position = poScreenCenter
OnShow = FormShow
PixelsPerInch = 96
TextHeight = 13
<-------> object DBChart1: TDBChart
Left = 145
Top = 0
Width = 455
Height = 439
AnimatedZoom = True
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
BackWall.Color = clWhite
BottomWall.Color = 16777088
Foot.Brush.Color = clBlue
Foot.Color = clBlue
Foot.Font.Charset = DEFAULT_CHARSET
Foot.Font.Color = clLime
Foot.Font.Height = -13
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsBold, fsItalic]
Foot.Frame.Color = clWhite
Foot.Frame.Width = 5
Foot.Frame.Visible = True
Foot.Text.Strings = (
'Who has all of them ?')
Title.Alignment = taLeftJustify
Title.Brush.Color = clNavy
Title.Color = clNavy
Title.Font.Charset = DEFAULT_CHARSET
Title.Font.Color = clYellow
Title.Font.Height = -19
Title.Font.Name = 'Arial'
Title.Font.Style = [fsBold, fsItalic]
Title.Frame.Color = clBlue
Title.Frame.Width = 5
Title.Frame.Visible = True
Title.Text.Strings = (
'Animal Comparison'

TCheckBox
Left = 15
Top = 44
Width = 97
Height = 17
Caption = 'C&ross Cursor'
Checked = True
State = cbChecked
TabOrder = 1
OnClick = CheckBox1Click
end
<-------> object BitBtn2: TBitBtn
Left = 11
Top = 260
Width = 118
Height = 33
Caption = 'CrossHair C&olor...'
TabOrder = 2
OnClick = BitBtn2Click
end
<-------> object Memo1: TMemo
Left = 8
Top = 80
Width = 121
Height = 161
Lines.Strings = (
'Cross-hair lines are '
'drawn over a TChart '
'using the '
'OnMouseMove event.'
''
'Coordinates for the '
'cross-hair lines are '
'calculated using the '
'Series GetCursorValues '
'method.')
TabOrder = 3
end
end
End
\\\
19
<-------> object DBHorizBarForm: TDBHorizBarForm
Left = 185
Top = 109
Width = 608
Height = 473

DEFAULT_CHARSET
TopAxis.LabelsFont.Color = clRed
TopAxis.LabelsFont.Height = -13
TopAxis.LabelsFont.Name = 'Arial'
TopAxis.LabelsFont.Style = [fsItalic]
TopAxis.Title.Caption = 'Weight in Kg'
TopAxis.Title.Font.Charset = DEFAULT_CHARSET
TopAxis.Title.Font.Color = clBlue
TopAxis.Title.Font.Height = -16
TopAxis.Title.Font.Name = 'Arial'
TopAxis.Title.Font.Style = []
Align = alClient
TabOrder = 0
<-------> object HorizBarSeries1: THorizBarSeries
HorizAxis = aTopAxis
Marks.Arrow.Color = clLime
Marks.ArrowLength = 20
Marks.BackColor = clWhite
Marks.Font.Charset = DEFAULT_CHARSET
Marks.Font.Color = clRed
Marks.Font.Height = -13
Marks.Font.Name = 'Arial'
Marks.Font.Style = [fsBold]
Marks.Style = smsXValue
Marks.Visible = True
DataSource = Table1
SeriesColor = clRed
Title = 'Weight'
XLabelsSource = 'NAME'
MultiBar = mbNone
XValues.DateTime = False
XValues.Name = 'Bar'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loNone
XValues.ValueSource = 'WEIGHT'
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone

'Size vs. Weight')
BackColor = clWhite
BottomAxis.LabelsFont.Charset = DEFAULT_CHARSET
BottomAxis.LabelsFont.Color = clGreen
BottomAxis.LabelsFont.Height = -13
BottomAxis.LabelsFont.Name = 'Arial'
BottomAxis.LabelsFont.Style = [fsBold, fsItalic]
BottomAxis.LabelStyle = talValue
BottomAxis.Title.Caption = 'Size in Inches'
BottomAxis.Title.Font.Charset = DEFAULT_CHARSET
BottomAxis.Title.Font.Color = clYellow
BottomAxis.Title.Font.Height = -16
BottomAxis.Title.Font.Name = 'Arial'
BottomAxis.Title.Font.Style = [fsBold]
Chart3DPercent = 50
LeftAxis.Inverted = True
LeftAxis.LabelsFont.Charset = DEFAULT_CHARSET
LeftAxis.LabelsFont.Color = clNavy
LeftAxis.LabelsFont.Height = -13
LeftAxis.LabelsFont.Name = 'Arial'
LeftAxis.LabelsFont.Style = [fsBold]
LeftAxis.LabelsSeparation = 5
LeftAxis.LabelStyle = talText
LeftAxis.Title.Caption = 'Animal Name'
LeftAxis.Title.Font.Charset = DEFAULT_CHARSET
LeftAxis.Title.Font.Color = clRed
LeftAxis.Title.Font.Height = -19
LeftAxis.Title.Font.Name = 'Arial'
LeftAxis.Title.Font.Style = [fsBold, fsItalic]
Legend.ColorWidth = 60
TopAxis.LabelsFont.Charset =

```
    State = cbChecked
    TabOrder = 0
    OnClick = CheckBox1Click
  end
  <-------> object RadioGroup1:
TRadioGroup
    Left = 6
    Top = 44
    Width = 123
    Height = 101
    Caption = 'MultiBar Style'
    ItemIndex = 0
    Items.Strings = (
      '&None'
      'S&ide to Side'
      '&Stacked'
      'S&tacked 100%')
    TabOrder = 1
    OnClick = RadioGroup1Click
  end
  <-------> object BitBtn2: TBitBtn
    Left = 22
    Top = 400
    Width = 95
    Height = 35
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 2
    Kind = bkClose
  end
  <-------> object BitBtn3: TBitBtn
    Left = 8
    Top = 359
    Width = 121
    Height = 34
    Caption = 'Swap Series'
    TabOrder = 3
    OnClick = BitBtn3Click
    Glyph.Data = {
      0303}
    NumGlyphs = 2
  end
  <-------> object CheckBox2:
```

```
    end
    <-------> object HorizBarSeries2:
THorizBarSeries
      Marks.Arrow.Color = clYellow
      Marks.ArrowLength = 20
      Marks.BackColor = clWhite
      Marks.Font.Charset =
DEFAULT_CHARSET
      Marks.Font.Color = clGreen
      Marks.Font.Height = -13
      Marks.Font.Name = 'Arial'
      Marks.Font.Style = [fsBold,
fsItalic]
      Marks.Style = smsXValue
      Marks.Visible = True
      DataSource = Table1
      SeriesColor = clBlue
      Title = 'Size'
      XLabelsSource = 'NAME'
      MultiBar = mbNone
      XValues.DateTime = False
      XValues.Name = 'Bar'
      XValues.Multiplier =
1.000000000000000000
      XValues.Order = loNone
      XValues.ValueSource = 'SIZE'
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier =
1.000000000000000000
      YValues.Order = loNone
    end
  end
  <-------> object Panel1: TPanel
    Left = 0
    Top = 0
    Width = 145
    Height = 439
    Align = alLeft
    TabOrder = 1
    <-------> object CheckBox1:
TCheckBox
      Left = 42
      Top = 155
      Width = 47
      Height = 17
      Caption = '3&D'
      Checked = True
```

Left = 382
Top = 17
end
End
\\\\\\\\\\\\\\\\\\\\
20
<-------> object DemoForm: TDemoForm
Left = 189
Top = 110
Width = 628
Height = 440
ActiveControl = ListBox1
Caption = 'DemoForm'
Color = clNavy
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
OldCreateOrder = True
Position = poScreenCenter
OnCreate = FormCreate
OnShow = FormShow
PixelsPerInch = 96
TextHeight = 13
<-------> object Label4: TLabel
Left = 121
Top = 260
Width = 23
Height = 13
HelpType = htKeyword
Alignment = taRightJustify
Caption = 'Slow'
end
<-------> object Label5: TLabel
Left = 12
Top = 260
Width = 20
Height = 13
HelpType = htKeyword
Caption = 'Fast'
end
<-------> object Panel2: TPanel
Left = 0
Top = 367
Width = 620

TCheckBox
Left = 8
Top = 8
Width = 137
Height = 25
Caption = '&Animate !!!'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -19
Font.Name = 'Arial'
Font.Style = [fsBold]
ParentFont = False
TabOrder = 4
OnClick = CheckBox2Click
end
<-------> object Memo1: TMemo
Left = 8
Top = 184
Width = 121
Height = 161
Lines.Strings = (
'Connecting a Series to '
'a TTable component is '
'just at few mouse clicks '
'using the Chart editor '
'dialog !'
''
'In this example 2 fields '
'from ANIMALS.DBF '
'table are connected to '
'2 Horizontal Bar'
'Series.')
TabOrder = 5
end
end
<-------> object Table1: TTable
Active = True
DatabaseName = 'DBDEMOS'
ReadOnly = True
TableName = 'ANIMALS.DBF'
Left = 431
Top = 18
end
<-------> object Timer1: TTimer
Enabled = False
Interval = 1
OnTimer = Timer1Timer

```
    Width = 85
    Height = 25
    Caption = 'Zoom Out'
    TabOrder = 2
    OnClick = BitBtn2Click
    Glyph.Data = {

333333773FF7733333333333370007333
3333333333777333333333}
    NumGlyphs = 2
   end
   <-------> object Button3: TButton
    Left = 204
    Top = 7
    Width = 75
    Height = 25
    Caption = 'Reset Zoom'
    Enabled = False
    TabOrder = 3
    OnClick = Button3Click
   end
  end
  <-------> object Panel3: TPanel
   Left = 453
   Top = 0
   Width = 167
   Height = 367
   Align = alRight
   TabOrder = 2
   <-------> object Image1: TImage
    Left = 14
    Top = 6
    Width = 141
    Height = 45
    HelpType = htKeyword
    AutoSize = True
    Picture.Data = {

07544269746D6170861D0000424D86
1D00000000000036040000280000008
D00

00002D0000000100080000000000501
90000000000000000000000001000000
01

00000000000080808000000008000008
08000008000008080000080000000080
00
   Height = 39
   Align = alBottom
   TabOrder = 1
   <-------> object SpeedButton1:
TSpeedButton
    Left = 312
    Top = 7
    Width = 25
    Height = 25
    HelpType = htKeyword
    Glyph.Data = {
    NumGlyphs = 2
    OnClick = SpeedButton1Click
   end
   <-------> object SpeedButton2:
TSpeedButton
    Left = 344
    Top = 7
    Width = 25
    Height = 25
    HelpType = htKeyword
    Glyph.Data = {
    NumGlyphs = 2
    OnClick = SpeedButton4Click
   end
   <-------> object Button2: TButton
    Left = 492
    Top = 7
    Width = 75
    Height = 25
    Caption = '< Back'
    TabOrder = 0
    OnClick = Button2Click
   end
   <-------> object BitBtn1: TBitBtn
    Left = 16
    Top = 7
    Width = 81
    Height = 25
    Caption = 'Zoom In'
    TabOrder = 1
    OnClick = BitBtn1Click
    Glyph.Data = {
    NumGlyphs = 2
   end
   <-------> object BitBtn2: TBitBtn
    Left = 110
    Top = 7
```

Width = 43
Height = 13
HelpType = htKeyword
Caption = '&Rotation:'
FocusControl = ScrollBar3
end
<-------> object ListBox1: TListBox
Left = 12
Top = 82
Width = 145
Height = 152
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ItemHeight = 13
Items.Strings = (
'Line / Strip'
'Bar / Stacked'
'Horizontal Bar /Stacked'
'Area / Step'
'Point XY / Scatter'
'Pie'
'Fast Line'
'Arrow'
'Bubble'
'Gantt'
'Shapes')
ParentFont = False
TabOrder = 0
OnClick = ListBox1Click
end
<-------> object CheckBox1: TCheckBox
Left = 53
Top = 292
Width = 37
Height = 17
Caption = '3&D:'
Checked = True
State = cbChecked
TabOrder = 1
OnClick = CheckBox1Click
end
<-------> object CheckBox2:

OnClick = Image1Click
end
<-------> object Label1: TLabel
Left = 12
Top = 66
Width = 64
Height = 13
HelpType = htKeyword
Caption = '&Series Types:'
FocusControl = ListBox1
end
<-------> object Label2: TLabel
Left = 17
Top = 247
Width = 20
Height = 13
HelpType = htKeyword
Caption = 'Fast'
end
<-------> object Label3: TLabel
Left = 126
Top = 247
Width = 23
Height = 13
HelpType = htKeyword
Alignment = taRightJustify
Caption = 'Slow'
end
<-------> object Label6: TLabel
Left = 17
Top = 300
Width = 14
Height = 13
HelpType = htKeyword
Caption = '2D'
end
<-------> object Label7: TLabel
Left = 135
Top = 300
Width = 14
Height = 13
HelpType = htKeyword
Alignment = taRightJustify
Caption = '3D'
end
<-------> object Label8: TLabel
Left = 8
Top = 346

```
    TabOrder = 5
    OnChange = ScrollBar3Change
  end
 end
 <-------> object Notebook1:
TNotebook
  Left = 0
  Top = 0
  Width = 453
  Height = 367
  Align = alClient
  TabOrder = 0
  <-------> object TPage
    Left = 0
    Top = 0
    Caption = 'Line'
    <-------> object Chart1: TChart
      Left = 0
      Top = 0
      Width = 453
      Height = 367
      BackWall.Brush.Color =
clWhite
      BackWall.Brush.Style =
bsClear
      BackWall.Color = clWhite
      BottomWall.Color = 16777088
      BottomWall.Size = 9
      LeftWall.Color = 8454016
      LeftWall.Size = 9
      Title.Brush.Color = 8388863
      Title.Color = 8388863
      Title.Font.Charset =
DEFAULT_CHARSET
      Title.Font.Color = clWhite
      Title.Font.Height = -19
      Title.Font.Name = 'Arial'
      Title.Font.Style = [fsBold,
fsItalic]
      Title.Frame.Width = 3
      Title.Frame.Visible = True
      Title.Text.Strings = (
        'Worldwide Fuel
Consumption'
        '( in millions of liters )')
      BackColor = clWhite
      BottomAxis.Axis.Color = clBlue
      BottomAxis.Grid.Visible =
False
```

```
TCheckBox
    Left = 53
    Top = 239
    Width = 57
    Height = 17
    Caption = '&Animate'
    Checked = True
    State = cbChecked
    TabOrder = 2
    OnClick = CheckBox2Click
  end
  <-------> object ScrollBar1:
TScrollBar
    Left = 17
    Top = 263
    Width = 133
    Height = 16
    LargeChange = 100
    Max = 1000
    Min = 1
    PageSize = 0
    Position = 1
    SmallChange = 25
    TabOrder = 3
    OnChange = ScrollBar1Change
  end
  <-------> object ScrollBar2:
TScrollBar
    Left = 17
    Top = 316
    Width = 133
    Height = 16
    Min = 1
    PageSize = 0
    Position = 15
    TabOrder = 4
    OnChange = ScrollBar2Change
  end
  <-------> object ScrollBar3:
TScrollBar
    Left = 56
    Top = 344
    Width = 94
    Height = 16
    Max = 360
    Min = 270
    PageSize = 0
    Position = 360
```

Legend.Font.Style = [fsBold]
Align = alClient
Color = 8454143
TabOrder = 0
<-------> object Series1: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = 65408
Title = 'Europe'
Pointer.Brush.Color = clRed
Pointer.HorizSize = 6
Pointer.InflateMargins = False
Pointer.Style = psCircle
Pointer.VertSize = 6
Pointer.Visible = True
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
<-------> object Series2: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clAqua
Title = 'America'
Pointer.Brush.Color = clLime
Pointer.HorizSize = 6
Pointer.InflateMargins = False
Pointer.Style = psTriangle
Pointer.VertSize = 6
Pointer.Visible = True
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000

BottomAxis.LabelsFont.Charset = DEFAULT_CHARSET
BottomAxis.LabelsFont.Color = clBlack
BottomAxis.LabelsFont.Height = -11
BottomAxis.LabelsFont.Name = 'Courier New'
BottomAxis.LabelsFont.Style = [fsBold]
BottomAxis.TickInnerLength = 6
BottomAxis.TicksInner.Width = 2
BottomAxis.Title.Caption = 'Measure Station'
BottomAxis.Title.Font.Charset = DEFAULT_CHARSET
BottomAxis.Title.Font.Color = clBlue
BottomAxis.Title.Font.Height = -13
BottomAxis.Title.Font.Name = 'Arial'
BottomAxis.Title.Font.Style = [fsBold]
LeftAxis.Grid.Color = clBlue
LeftAxis.MinorTickCount = 2
LeftAxis.MinorTickLength = 4
LeftAxis.TickInnerLength = 9
LeftAxis.TickLength = 7
LeftAxis.Ticks.Width = 2
LeftAxis.Title.Caption = 'Liters'
LeftAxis.Title.Font.Charset = DEFAULT_CHARSET
LeftAxis.Title.Font.Color = clNavy
LeftAxis.Title.Font.Height = -16
LeftAxis.Title.Font.Name = 'Arial'
LeftAxis.Title.Font.Style = [fsBold, fsItalic]
Legend.Color = clBlue
Legend.ColorWidth = 27
Legend.Font.Charset = DEFAULT_CHARSET
Legend.Font.Color = clWhite
Legend.Font.Height = -13
Legend.Font.Name = 'Arial'

end
end
<-------> object TPage
Left = 0
Top = 0
Caption = 'Bar'
<-------> object Chart2: TChart
Left = 0
Top = 0
Width = 453
Height = 367
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
BackWall.Color = 8454143
BackWall.Pen.Color = clRed
BottomWall.Size = 4
LeftWall.Color = 16777088
LeftWall.Size = 4
Title.Alignment = taRightJustify
Title.Font.Charset = DEFAULT_CHARSET
Title.Font.Color = clBlue
Title.Font.Height = -16
Title.Font.Name = 'Arial'
Title.Font.Style = []
Title.Text.Strings = (
'American Charting Suppliers, Inc.'
'1997 Production Volume')
BackColor = 8454143
Frame.Color = clRed
LeftAxis.Grid.Visible = False
LeftAxis.Title.Caption = 'Millions of $'
LeftAxis.Title.Font.Charset = DEFAULT_CHARSET
LeftAxis.Title.Font.Color = clBlack
LeftAxis.Title.Font.Height = -15
LeftAxis.Title.Font.Name = 'Arial'
LeftAxis.Title.Font.Style = []
Legend.Alignment = laTop
Legend.ColorWidth = 22
Legend.DividingLines.Color = clSilver

YValues.Order = loNone
end
<-------> object Series3: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clYellow
Title = 'Asia'
LinePen.Visible = False
Pointer.Brush.Color = clBlue
Pointer.HorizSize = 6
Pointer.InflateMargins = False
Pointer.Style = psDownTriangle
Pointer.VertSize = 6
Pointer.Visible = True
Stairs = True
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
<-------> object Series4: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clBlue
Title = 'Africa'
Pointer.Brush.Color = 8454143
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = True
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end

```
        XValues.DateTime = True
        XValues.Name = 'X'
        XValues.Multiplier =
1.000000000000000000
        XValues.Order = loAscending
        YValues.DateTime = False
        YValues.Name = 'Bar'
        YValues.Multiplier =
1.000000000000000000
        YValues.Order = loNone
      end
      <-------> object Series6:
TBarSeries
        Marks.ArrowLength = 20
        Marks.BackColor = 65408
        Marks.Visible = True
        SeriesColor = clBlue
        Title = 'Paris'
        MultiBar = mbNone
        XValues.DateTime = True
        XValues.Name = 'X'
        XValues.Multiplier =
1.000000000000000000
        XValues.Order = loAscending
        YValues.DateTime = False
        YValues.Name = 'Bar'
        YValues.Multiplier =
1.000000000000000000
        YValues.Order = loNone
      end
    end
  end
  <-------> object TPage
    Left = 0
    Top = 0
    Caption = 'Horizontal Bar'
    <-------> object Chart3: TChart
      Left = 0
      Top = 0
      Width = 453
      Height = 367
      BackWall.Brush.Color =
clWhite
      BackWall.Brush.Style =
bsClear
      BackWall.Color = 12615935
      BottomWall.Size = 8
      Foot.Font.Charset =
DEFAULT_CHARSET
```

```
      Legend.DividingLines.Visible =
True
      Legend.ShadowColor = clGray
      Legend.ShadowSize = 5
      Align = alClient
      TabOrder = 0
      <-------> object ComboBox1:
TComboBox
        Left = 12
        Top = 8
        Width = 109
        Height = 21
        Style = csDropDownList
        ItemHeight = 13
        TabOrder = 0
        OnChange =
ComboBox1Change
        Items.Strings = (
          'None'
          'Side to Side'
          'Stacked'
          'Stacked 100%')
      end
      <-------> object Series5:
TBarSeries
        Marks.ArrowLength = 20
        Marks.Visible = False
        SeriesColor = clNavy
        Title = 'San Francisco'
        MultiBar = mbNone
        XValues.DateTime = False
        XValues.Name = 'X'
        XValues.Multiplier =
1.000000000000000000
        XValues.Order = loAscending
        YValues.DateTime = False
        YValues.Name = 'Bar'
        YValues.Multiplier =
1.000000000000000000
        YValues.Order = loNone
      end
      <-------> object Series7:
TBarSeries
        Marks.ArrowLength = 20
        Marks.BackColor = 16777088
        Marks.Visible = False
        SeriesColor = clYellow
        Title = 'Hong Kong'
        MultiBar = mbNone
```

THorizBarSeries
Marks.ArrowLength = 20
Marks.Visible = False
SeriesColor = clRed
Title = 'IP'
VertAxis = aRightAxis
MultiBar = mbStacked
XValues.DateTime = False
XValues.Name = 'Bar'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loNone
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
<-------> object Series10: THorizBarSeries
Marks.ArrowLength = 20
Marks.Visible = False
SeriesColor = clYellow
Title = 'Netware'
VertAxis = aRightAxis
MultiBar = mbStacked
XValues.DateTime = False
XValues.Name = 'Bar'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loNone
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
<-------> object Series9: THorizBarSeries
Marks.ArrowLength = 20
Marks.BackColor = clBlue
Marks.Font.Charset = DEFAULT_CHARSET
Marks.Font.Color = clAqua
Marks.Font.Height = -11
Marks.Font.Name = 'Arial'
Marks.Font.Style = []
Marks.Frame.Color = clWhite
Marks.Visible = True

Foot.Font.Color = clRed
Foot.Font.Height = -19
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsItalic]
Foot.Text.Strings = (
'Ethernet Protocol Statistics')
LeftWall.Size = 8
MarginLeft = 4
MarginRight = 4
MarginTop = 9
Title.Text.Strings = (
'')
Title.Visible = False
BackColor = 12615935
Chart3DPercent = 40
Legend.Alignment = laTop
Legend.TopPos = 19
RightAxis.Title.Caption = 'Hub'
TopAxis.Title.Caption = 'Packets (in thousands)'
TopAxis.Title.Font.Charset = DEFAULT_CHARSET
TopAxis.Title.Font.Color = clNavy
TopAxis.Title.Font.Height = -15
TopAxis.Title.Font.Name = 'Arial'
TopAxis.Title.Font.Style = [fsBold, fsItalic]
Align = alClient
TabOrder = 0
<-------> object ComboBox2: TComboBox
Left = 12
Top = 8
Width = 109
Height = 21
Style = csDropDownList
ItemHeight = 13
TabOrder = 0
OnChange = ComboBox1Change
Items.Strings = (
'None'
'Side to Side'
'Stacked'
'Stacked 100%')
end
<-------> object Series8:

BottomAxis.Grid.Style = psSolid
Chart3DPercent = 30
LeftAxis.Grid.Style = psSolid
Legend.Color = 8454143
Legend.Font.Charset = DEFAULT_CHARSET
Legend.Font.Color = clBlack
Legend.Font.Height = -12
Legend.Font.Name = 'Arial'
Legend.Font.Style = [fsBold]
Legend.Frame.Width = 2
RightAxis.Grid.Color = clYellow
RightAxis.Grid.Style = psSolid
RightAxis.LabelsFont.Charset = DEFAULT_CHARSET
RightAxis.LabelsFont.Color = clBlack
RightAxis.LabelsFont.Height = -11
RightAxis.LabelsFont.Name = 'Arial'
RightAxis.LabelsFont.Style = [fsBold]
RightAxis.Title.Caption = 'Price'
TopAxis.Grid.Style = psSolid
TopAxis.Grid.Visible = False
TopAxis.LabelsFont.Charset = DEFAULT_CHARSET
TopAxis.LabelsFont.Color = clPurple
TopAxis.LabelsFont.Height = -12
TopAxis.LabelsFont.Name = 'Arial'
TopAxis.LabelsFont.Style = []
Align = alClient
TabOrder = 0
<-------> object CheckBox4: TCheckBox
Left = 8
Top = 8
Width = 57
Height = 17
Caption = 'Steps'
TabOrder = 0
OnClick = CheckBox4Click

SeriesColor = 8454016
Title = 'ARP'
VertAxis = aRightAxis
MultiBar = mbStacked
XValues.DateTime = False
XValues.Name = 'Bar'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loNone
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
end
end
<-------> object TPage
Left = 0
Top = 0
Caption = 'Area'
<-------> object Chart4: TChart
Left = 0
Top = 0
Width = 453
Height = 374
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
BottomWall.Color = clNavy
BottomWall.Size = 8
Foot.Text.Strings = (
''
'number of users')
LeftWall.Color = clWhite
LeftWall.Size = 8
MarginBottom = 6
MarginLeft = 6
Title.Font.Charset = DEFAULT_CHARSET
Title.Font.Color = clBlue
Title.Font.Height = -13
Title.Font.Name = 'Arial'
Title.Font.Style = [fsBold]
Title.Text.Strings = (
'Database licences policy'
'Cost per user')

YValues.Order = loNone
end
end
end
<-------> object TPage
Left = 0
Top = 0
Caption = 'Point XY'
<-------> object Chart5: TChart
Left = 0
Top = 0
Width = 453
Height = 374
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
BackWall.Color = clGray
BottomWall.Color = clSilver
BottomWall.Size = 6
Gradient.Direction = gdRightLeft
Gradient.Visible = True
LeftWall.Color = clSilver
LeftWall.Size = 6
Title.Font.Charset = DEFAULT_CHARSET
Title.Font.Color = clTeal
Title.Font.Height = -16
Title.Font.Name = 'Arial'
Title.Font.Style = [fsBold]
Title.Text.Strings = (
'Fuel Consumption')
BackColor = clGray
BottomAxis.Grid.Visible = False
LeftAxis.Grid.Visible = False
LeftAxis.LabelsFont.Charset = DEFAULT_CHARSET
LeftAxis.LabelsFont.Color = clOlive
LeftAxis.LabelsFont.Height = -11
LeftAxis.LabelsFont.Name = 'Arial'
LeftAxis.LabelsFont.Style = [fsBold]
Legend.Alignment = laTop
Legend.Color = clBlack

end
<-------> object Series11: TAreaSeries
HorizAxis = aTopAxis
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clAqua
Title = 'SQL-Provider 1.2'
VertAxis = aRightAxis
AreaColor = clAqua
AreaLinesPen.Color = clGray
DrawArea = True
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
<-------> object Series12: TAreaSeries
HorizAxis = aTopAxis
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clTeal
Title = 'Super-OLAP 2.0'
VertAxis = aRightAxis
AreaColor = clTeal
DrawArea = True
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000

YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
<-------> object Series14: TPointSeries
Marks.ArrowLength = 0
Marks.Visible = False
SeriesColor = clGreen
Title = 'Bikes'
Pointer.HorizSize = 6
Pointer.InflateMargins = True
Pointer.Style = psCircle
Pointer.VertSize = 6
Pointer.Visible = True
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
<-------> object Series15: TPointSeries
Marks.ArrowLength = 0
Marks.Visible = False
SeriesColor = clYellow
Title = 'Trucks'
Pointer.HorizSize = 6
Pointer.InflateMargins = True
Pointer.Style = psTriangle
Pointer.VertSize = 6
Pointer.Visible = True
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end

Legend.ColorWidth = 32
Legend.Font.Charset = DEFAULT_CHARSET
Legend.Font.Color = clAqua
Legend.Font.Height = -12
Legend.Font.Name = 'Arial'
Legend.Font.Style = []
Legend.Frame.Color = clRed
Legend.Frame.Width = 2
Legend.ShadowColor = clWhite
Legend.ShadowSize = 1
RightAxis.Grid.Style = psSolid
RightAxis.LabelsFont.Charset = DEFAULT_CHARSET
RightAxis.LabelsFont.Color = clNavy
RightAxis.LabelsFont.Height = -12
RightAxis.LabelsFont.Name = 'Arial'
RightAxis.LabelsFont.Style = [fsItalic]
TopAxis.Grid.Style = psSolid
TopAxis.LabelsFont.Charset = DEFAULT_CHARSET
TopAxis.LabelsFont.Color = clBlue
TopAxis.LabelsFont.Height = -12
TopAxis.LabelsFont.Name = 'Arial'
TopAxis.LabelsFont.Style = []
Align = alClient
TabOrder = 0
<-------> object Series13: TPointSeries
Marks.ArrowLength = 0
Marks.Visible = False
SeriesColor = clRed
Title = 'Cars'
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = True
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False

```
      HorizAxis = aTopAxis
      Marks.ArrowLength = 0
      Marks.Visible = False
      SeriesColor = clGray
      Title = 'Rockets'
      VertAxis = aRightAxis
      Pointer.InflateMargins = True
      Pointer.Style = psDiamond
      Pointer.VertSize = 9
      Pointer.Visible = True
      XValues.DateTime = False
      XValues.Name = 'X'
      XValues.Multiplier = 1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier = 1.000000000000000000
      YValues.Order = loNone
    end
  end
end
<-------> object TPage
  Left = 0
  Top = 0
  Caption = 'Pie'
  <-------> object Chart6: TChart
    Left = 0
    Top = 0
    Width = 453
    Height = 374
    AllowPanning = pmNone
    AllowZoom = False
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    Foot.Text.Strings = (
      'The Furniture Specialists Intl.')
    Gradient.Direction = gdRightLeft
    Gradient.Visible = True
    Title.Font.Charset = DEFAULT_CHARSET
    Title.Font.Color = clBlue
    Title.Font.Height = -16
```

```
    <-------> object Series16: TPointSeries
      HorizAxis = aTopAxis
      Marks.ArrowLength = 0
      Marks.Visible = False
      SeriesColor = clBlue
      Title = 'Boats'
      VertAxis = aRightAxis
      Pointer.HorizSize = 8
      Pointer.InflateMargins = True
      Pointer.Style = psDownTriangle
      Pointer.VertSize = 8
      Pointer.Visible = True
      XValues.DateTime = False
      XValues.Name = 'X'
      XValues.Multiplier = 1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier = 1.000000000000000000
      YValues.Order = loNone
    end
    <-------> object Series17: TPointSeries
      Marks.ArrowLength = 0
      Marks.Visible = False
      SeriesColor = clWhite
      Title = 'Planes'
      Pointer.HorizSize = 7
      Pointer.InflateMargins = True
      Pointer.Style = psDiagCross
      Pointer.VertSize = 2
      Pointer.Visible = True
      XValues.DateTime = False
      XValues.Name = 'X'
      XValues.Multiplier = 1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier = 1.000000000000000000
      YValues.Order = loNone
    end
    <-------> object Series18: TPointSeries
```

```
      Marks.Visible = True
      SeriesColor = clRed
      OtherSlice.Text = 'Other'
      PieValues.DateTime = False
      PieValues.Name = 'Pie'
      PieValues.Multiplier = 1.000000000000000000
      PieValues.Order = loNone
    end
  end
end
<-------> object TPage
  Left = 0
  Top = 0
  Caption = 'Fast Line'
  <-------> object Chart7: TChart
    Left = 0
    Top = 0
    Width = 453
    Height = 374
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    Title.AdjustFrame = False
    Title.Brush.Color = clBlue
    Title.Font.Charset = DEFAULT_CHARSET
    Title.Font.Color = clAqua
    Title.Font.Height = -19
    Title.Font.Name = 'Arial'
    Title.Font.Style = []
    Title.Frame.Width = 2
    Title.Frame.Visible = True
    Title.Text.Strings = (
      'Traded Volume from 1990 to 1996')
    BottomAxis.Title.Caption = 'Trading Sessions'
    LeftAxis.Title.Caption = 'Number of Shares'
    Legend.Alignment = laTop
    Legend.Color = clGray
    Legend.Font.Charset = DEFAULT_CHARSET
    Legend.Font.Color = clWhite
    Legend.Font.Height = -11
    Legend.Font.Name = 'Arial'
    Legend.Font.Style = []
```

```
    Title.Font.Name = 'Arial'
    Title.Font.Style = []
    Title.Text.Strings = (
      '% Growth Rate ')
    AxisVisible = False
    ClipPoints = False
    Frame.Visible = False
    Legend.Alignment = laLeft
    Legend.Color = 8453888
    Legend.Frame.Color = clBlue
    Legend.Frame.Style = psDot
    Legend.Frame.Width = 2
    Legend.ShadowColor = clGray
    Legend.ShadowSize = 4
    Legend.TextStyle = ltsLeftPercent
    Legend.TopPos = 59
    View3DOptions.Elevation = 315
    View3DOptions.Orthogonal = False
    View3DOptions.Perspective = 0
    View3DOptions.Rotation = 360
    View3DWalls = False
    Align = alClient
    TabOrder = 0
    <-------> object CheckBox3: TCheckBox
      Left = 16
      Top = 20
      Width = 57
      Height = 17
      Caption = 'Circled'
      Color = clYellow
      ParentColor = False
      TabOrder = 0
      OnClick = CheckBox3Click
    end
    <-------> object Series19: TPieSeries
      Marks.Arrow.Color = clBlack
      Marks.ArrowLength = 8
      Marks.BackColor = clBlack
      Marks.Font.Charset = DEFAULT_CHARSET
      Marks.Font.Color = clWhite
      Marks.Font.Height = -11
      Marks.Font.Name = 'Arial'
      Marks.Font.Style = []
```

1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
<-------> object Series23: TFastLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clBlue
Title = 'Barcelona'
LinePen.Color = clBlue
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
<-------> object Series24: TFastLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clWhite
Title = 'Montreal'
LinePen.Color = clWhite
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
end
end
<-------> object TPage
Left = 0
Top = 0

View3D = False
Align = alClient
TabOrder = 0
<-------> object Series20: TFastLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clRed
Title = 'New York'
LinePen.Color = clRed
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
<-------> object Series21: TFastLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clGreen
Title = 'Chicago'
LinePen.Color = clGreen
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
end
<-------> object Series22: TFastLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clYellow
Title = 'Copenhague'
LinePen.Color = clYellow
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier =

EndYValues.Order = loNone
StartXValues.DateTime = False
StartXValues.Name = 'X'
StartXValues.Multiplier = 1.000000000000000000
StartXValues.Order = loAscending
StartYValues.DateTime = False
StartYValues.Name = 'Y'
StartYValues.Multiplier = 1.000000000000000000
StartYValues.Order = loNone
end
<-------> object Series27: TArrowSeries
Marks.ArrowLength = 0
Marks.Frame.Visible = False
Marks.Transparent = True
Marks.Visible = False
SeriesColor = clGreen
Pointer.InflateMargins = False
Pointer.Style = psRectangle
Pointer.Visible = True
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
EndXValues.DateTime = True
EndXValues.Name = 'EndX'
EndXValues.Multiplier = 1.000000000000000000
EndXValues.Order = loNone
EndYValues.DateTime = False
EndYValues.Name = 'EndY'
EndYValues.Multiplier = 1.000000000000000000
EndYValues.Order = loNone
StartXValues.DateTime = True
StartXValues.Name = 'X'
StartXValues.Multiplier =

Caption = 'Arrow'
<-------> object Chart8: TChart
Left = 0
Top = 0
Width = 453
Height = 374
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
Title.Font.Charset = DEFAULT_CHARSET
Title.Font.Color = clBlue
Title.Font.Height = -16
Title.Font.Name = 'Arial'
Title.Font.Style = []
Title.Text.Strings = (
'Weather in Japan')
Align = alClient
TabOrder = 0
<-------> object Series26: TArrowSeries
Marks.ArrowLength = 0
Marks.Frame.Visible = False
Marks.Transparent = True
Marks.Visible = False
SeriesColor = clRed
Pointer.InflateMargins = False
Pointer.Style = psRectangle
Pointer.Visible = True
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
EndXValues.DateTime = True
EndXValues.Name = 'EndX'
EndXValues.Multiplier = 1.000000000000000000
EndXValues.Order = loNone
EndYValues.DateTime = False
EndYValues.Name = 'EndY'
EndYValues.Multiplier = 1.000000000000000000

```
1.0000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier = 1.0000000000000000000
      YValues.Order = loNone
      RadiusValues.DateTime = False
      RadiusValues.Name = 'Radius'
      RadiusValues.Multiplier = 1.0000000000000000000
      RadiusValues.Order = loNone
    end
  end
end
<-------> object TPage
  Left = 0
  Top = 0
  Caption = 'Gantt'
  <-------> object Chart10: TChart
    Left = 0
    Top = 0
    Width = 453
    Height = 374
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    Title.Text.Strings = (
      'TChart')
    Chart3DPercent = 5
    Legend.Visible = False
    Align = alClient
    TabOrder = 0
    <-------> object Series29: TGanttSeries
      ColorEachPoint = True
      Marks.ArrowLength = 0
      Marks.Visible = False
      SeriesColor = clRed
      Pointer.InflateMargins = True
      Pointer.Style = psRectangle
      Pointer.Visible = True
      XValues.DateTime = True
      XValues.Name = 'Start'
      XValues.Multiplier = 1.0000000000000000000
      XValues.Order = loAscending
```

```
1.0000000000000000000
      StartXValues.Order = loAscending
      StartYValues.DateTime = False
      StartYValues.Name = 'Y'
      StartYValues.Multiplier = 1.0000000000000000000
      StartYValues.Order = loNone
    end
  end
end
<-------> object TPage
  Left = 0
  Top = 0
  Caption = 'Bubble'
  <-------> object Chart9: TChart
    Left = 0
    Top = 0
    Width = 453
    Height = 374
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    Title.Text.Strings = (
      'TChart')
    Legend.Visible = False
    Align = alClient
    TabOrder = 0
    <-------> object Series28: TBubbleSeries
      HorizAxis = aTopAxis
      Marks.ArrowLength = 0
      Marks.Frame.Visible = False
      Marks.Transparent = True
      Marks.Visible = True
      SeriesColor = clRed
      VertAxis = aRightAxis
      OnGetMarkText = Series28GetMarkText
      Pointer.HorizSize = 20
      Pointer.InflateMargins = False
      Pointer.Style = psCircle
      Pointer.VertSize = 20
      Pointer.Visible = True
      XValues.DateTime = False
      XValues.Name = 'X'
      XValues.Multiplier =
```

```
        Font.Color = clBlack
        Font.Height = -11
        Font.Name = 'Arial'
        Font.Style = []
        X0 = 20.000000000000000000
        X1 = 100.000000000000000000
        Y0 = 20.000000000000000000
        Y1 = 100.000000000000000000
        XValues.DateTime = False
        XValues.Name = 'X'
        XValues.Multiplier = 1.000000000000000000
        XValues.Order = loAscending
        YValues.DateTime = False
        YValues.Name = 'Y'
        YValues.Multiplier = 1.000000000000000000
        YValues.Order = loNone
      end
      <-------> object Series31: TChartShape
        Marks.ArrowLength = 8
        Marks.Visible = False
        SeriesColor = clWhite
        Brush.Color = clYellow
        Font.Charset = DEFAULT_CHARSET
        Font.Color = clBlack
        Font.Height = -11
        Font.Name = 'Arial'
        Font.Style = []
        Style = chasDiamond
        X0 = 30.000000000000000000
        X1 = 90.000000000000000000
        Y0 = 30.500000000000000000
        Y1 = 114.000000000000000000
        XValues.DateTime = False
        XValues.Name = 'X'
        XValues.Multiplier = 1.000000000000000000
        XValues.Order = loAscending
        YValues.DateTime = False
        YValues.Name = 'Y'
        YValues.Multiplier = 1.000000000000000000
        YValues.Order = loNone
      end
      <-------> object Series32: TChartShape
```

```
        YValues.DateTime = False
        YValues.Name = 'Y'
        YValues.Multiplier = 1.000000000000000000
        YValues.Order = loNone
        StartValues.DateTime = True
        StartValues.Name = 'Start'
        StartValues.Multiplier = 1.000000000000000000
        StartValues.Order = loAscending
        EndValues.DateTime = False
        EndValues.Name = 'End'
        EndValues.Multiplier = 1.000000000000000000
        EndValues.Order = loNone
        NextTask.DateTime = False
        NextTask.Name = 'NextTask'
        NextTask.Multiplier = 1.000000000000000000
        NextTask.Order = loNone
      end
    end
  end
  <-------> object TPage
    Left = 0
    Top = 0
    Caption = 'Shapes'
    <-------> object Chart11: TChart
      Left = 0
      Top = 0
      Width = 453
      Height = 374
      BackWall.Brush.Color = clWhite
      BackWall.Brush.Style = bsClear
      Title.Text.Strings = (
        'TChart')
      Align = alClient
      TabOrder = 0
      <-------> object Series30: TChartShape
        Marks.ArrowLength = 8
        Marks.Visible = False
        SeriesColor = clWhite
        Brush.Color = clWhite
        Font.Charset = DEFAULT_CHARSET
```

```
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 161
    Top = 0
    Width = 469
    Height = 404
    AnimatedZoom = True
    AnimatedZoomSteps = 16
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Color = 8454016
    BackWall.Pen.Color = clNavy
    BackWall.Pen.Width = 2
    BottomWall.Size = 6
    Foot.Frame.Color = clScrollBar
    LeftWall.Size = 4
    Title.Frame.Color = clScrollBar
    Title.Text.Strings = (
      'We''re testing that new Pentium...'
      'It seems has good health !!!')
    BackColor = 8454016
    BottomAxis.Grid.Color = 8388863
    Frame.Color = clNavy
    Frame.Width = 2
    LeftAxis.Grid.Color = clSilver
    Legend.Font.Charset = DEFAULT_CHARSET
    Legend.Font.Color = clBlack
    Legend.Font.Height = -19
    Legend.Font.Name = 'Arial'
    Legend.Font.Style = [fsBold, fsItalic]
    Legend.LegendStyle = lsLastValues
    OnGetLegendText = Chart1GetLegendText
    Align = alClient
    TabOrder = 0
    <-------> object LineSeries1: TLineSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
          Marks.ArrowLength = 8
          Marks.Visible = False
          SeriesColor = clWhite
          Brush.Color = clAqua
          Font.Charset = DEFAULT_CHARSET
          Font.Color = clBlack
          Font.Height = -11
          Font.Name = 'Arial'
          Font.Style = []
          Style = chasTriangle
          X0 = 30.000000000000000000
          X1 = 90.000000000000000000
          Y0 = 47.500000000000000000
          Y1 = 102.000000000000000000
          XValues.DateTime = False
          XValues.Name = 'X'
          XValues.Multiplier = 1.000000000000000000
          XValues.Order = loAscending
          YValues.DateTime = False
          YValues.Name = 'Y'
          YValues.Multiplier = 1.000000000000000000
          YValues.Order = loNone
        end
      end
    end
  end
  <-------> object Timer1: TTimer
    Enabled = False
    Interval = 300
    OnTimer = Timer1Timer
    Left = 404
    Top = 18
  end
End
\\\\\\\\\\\\\\\\\\\\\\\
21
<-------> object DigitalForm: TDigitalForm
  Left = 196
  Top = 113
  Width = 638
  Height = 438
  Caption = 'TeeChart Digital Series Example'
  Color = clBtnFace
```

Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
Stairs = True
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 107
Top = 86
end
<-------> object LineSeries4: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clRed
Title = 'I/O Timing'
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
Stairs = True
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 173
Top = 88
end
end
<-------> object Panel1: TPanel
Left = 0
Top = 0
Width = 161
Height = 404
Align = alLeft
TabOrder = 1

SeriesColor = clBlue
Title = 'Pipeline Signal'
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
Stairs = True
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 175
Top = 130
end
<-------> object LineSeries2: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clYellow
Title = 'RAM Cycles'
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
Stairs = True
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 105
Top = 132
end
<-------> object LineSeries3: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clGreen
Title = 'PCI Status'

```
    Kind = bkClose
  end
  <-------> object CheckBox4:
TCheckBox
    Left = 20
    Top = 93
    Width = 78
    Height = 18
    Caption = 'Z Order'
    TabOrder = 4
    OnClick = CheckBox4Click
  end
  <-------> object Memo1: TMemo
    Left = 20
    Top = 136
    Width = 121
    Height = 169
    Lines.Strings = (
      'Line Series can be used '
      'as "step" Series by '
      'setting '
      'the Stairs property.'
      ''
      'In this demo several '
      'Line Series are '
      'displayed '
      'on same Z order.')
    TabOrder = 5
  end
 end
 <-------> object Timer1: TTimer
  Enabled = False
  Interval = 1
  OnTimer = Timer1Timer
  Left = 170
  Top = 10
 end
End
\\\\\\\\\\\\\
22
<-------> object DrawForm:
TDrawForm
 Left = 182
 Top = 109
 Width = 599
 Height = 398
 Caption = 'TeeChart Customized
Drawing Example'
```

```
  <-------> object CheckBox1:
TCheckBox
    Left = 20
    Top = 13
    Width = 119
    Height = 18
    Caption = 'Animate !!!'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -19
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 0
    OnClick = CheckBox1Click
  end
  <-------> object CheckBox2:
TCheckBox
    Left = 20
    Top = 69
    Width = 78
    Height = 18
    Caption = 'View 3D'
    Checked = True
    State = cbChecked
    TabOrder = 1
    OnClick = CheckBox2Click
  end
  <-------> object CheckBox3:
TCheckBox
    Left = 20
    Top = 45
    Width = 126
    Height = 18
    Caption = 'Last Values in
Legend'
    Checked = True
    State = cbChecked
    TabOrder = 2
    OnClick = CheckBox3Click
  end
  <-------> object BitBtn3: TBitBtn
    Left = 32
    Top = 351
    Width = 89
    Height = 33
    TabOrder = 3
```

```
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier =
1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier =
1.000000000000000000
    YValues.Order = loNone
    Left = 199
    Top = 122
  end
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 0
  Width = 145
  Height = 364
  Align = alLeft
  TabOrder = 1
  <-------> object BitBtn3: TBitBtn
    Left = 24
    Top = 312
    Width = 89
    Height = 33
    TabOrder = 0
    Kind = bkClose
  end
  <-------> object CheckBox1:
TCheckBox
    Left = 16
    Top = 12
    Width = 117
    Height = 17
    Caption = '&Animate !!!'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -19
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 1
    OnClick = CheckBox1Click
  end
  <-------> object BitBtn1: TBitBtn
    Left = 16
```

```
 Color = clBtnFace
 ParentFont = True
 OldCreateOrder = True
 Position = poScreenCenter
 OnCreate = FormCreate
 PixelsPerInch = 96
 TextHeight = 13
 <-------> object Chart1: TChart
   Left = 145
   Top = 0
   Width = 446
   Height = 364
   BackWall.Brush.Color = clWhite
   BackWall.Brush.Style = bsClear
   BackWall.Color = clSilver
   BottomWall.Color = 16777088
   BottomWall.Size = 4
   Foot.Font.Charset =
DEFAULT_CHARSET
   Foot.Font.Color = clRed
   Foot.Font.Height = -12
   Foot.Font.Name = 'Arial'
   Foot.Font.Style = [fsItalic]
   Foot.Frame.Color = clScrollBar
   LeftWall.Color = 8454016
   LeftWall.Size = 4
   Title.Frame.Color = clScrollBar
   Title.Text.Strings = (
     'TChart Component')
   BackColor = clSilver
   Chart3DPercent = 65
   Legend.ColorWidth = 32
   Align = alClient
   Color = clWhite
   TabOrder = 0
   <-------> object LineSeries1:
TLineSeries
     Marks.ArrowLength = 8
     Marks.Visible = False
     SeriesColor = clBlue
     AfterDrawValues =
LineSeries1AfterDrawValues
     BeforeDrawValues =
LineSeries1BeforeDrawValues
     LinePen.Width = 3
     Pointer.InflateMargins = True
     Pointer.Style = psRectangle
     Pointer.Visible = False
```

```
TFastLineForm
  Left = 192
  Top = 108
  Width = 575
  Height = 415
  Caption = 'TeeChart Fast
LineSeries Example'
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 153
    Top = 0
    Width = 414
    Height = 381
    AnimatedZoom = True
    AnimatedZoomSteps = 15
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Color = clBlue
    BackWall.Pen.Visible = False
    Foot.Font.Charset =
DEFAULT_CHARSET
    Foot.Font.Color = clRed
    Foot.Font.Height = -12
    Foot.Font.Name = 'Arial'
    Foot.Font.Style = [fsItalic]
    Foot.Frame.Color = clScrollBar
    Foot.Visible = False
    MarginBottom = 1
    MarginLeft = 1
    MarginRight = 1
    MarginTop = 1
    Title.Frame.Color = clScrollBar
    Title.Text.Strings = (
      'TChart Component')
    Title.Visible = False
    AxisVisible = False
    BottomAxis.Grid.Color = 8421440
    BottomAxis.LabelsFont.Charset =
DEFAULT_CHARSET
    BottomAxis.LabelsFont.Color =
clFuchsia
    BottomAxis.LabelsFont.Height = -
      Top = 40
      Width = 89
      Height = 33
      Caption = '&Print !'
      TabOrder = 2
      OnClick = BitBtn1Click
      Glyph.Data = {
      NumGlyphs = 2
    end
    <-------> object CheckBox2:
TCheckBox
      Left = 16
      Top = 84
      Width = 57
      Height = 17
      Caption = '3D'
      Checked = True
      State = cbChecked
      TabOrder = 3
      OnClick = CheckBox2Click
    end
    <-------> object Memo1: TMemo
      Left = 12
      Top = 116
      Width = 121
      Height = 169
      Lines.Strings = (
        'Custom drawing using '
        'the Chart Canvas '
        'property is easy !'
        ''
        'This demo shows how '
        'custom drawings can be '
        'printed too.')
      TabOrder = 4
    end
  end
  <-------> object Timer1: TTimer
    Enabled = False
    Interval = 1
    OnTimer = Timer1Timer
    Left = 642
    Top = 37
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
23
<-------> object FastLineForm:
```

Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clRed
VertAxis = aRightAxis
LinePen.Color = clRed
LinePen.Style = psDot
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 199
Top = 204
end
<-------> object FastLineSeries2: TFastLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clGreen
LinePen.Color = clGreen
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 237
Top = 206
end
end
<-------> object Panel1: TPanel
Left = 0
Top = 0
Width = 153
Height = 381
Align = alLeft
TabOrder = 1
<-------> object Button1: TButton
Left = 13

13
BottomAxis.LabelsFont.Name = 'Arial'
BottomAxis.LabelsFont.Style = []
BottomAxis.Ticks.Color = clLime
ClipPoints = False
Frame.Color = clBlue
Frame.Visible = False
LeftAxis.Grid.Color = clBlue
LeftAxis.LabelsFont.Charset = DEFAULT_CHARSET
LeftAxis.LabelsFont.Color = clNavy
LeftAxis.LabelsFont.Height = -13
LeftAxis.LabelsFont.Name = 'Arial'
LeftAxis.LabelsFont.Style = []
LeftAxis.Ticks.Color = clRed
Legend.Visible = False
RightAxis.Grid.Visible = False
RightAxis.LabelsFont.Charset = DEFAULT_CHARSET
RightAxis.LabelsFont.Color = clRed
RightAxis.LabelsFont.Height = -13
RightAxis.LabelsFont.Name = 'Arial'
RightAxis.LabelsFont.Style = []
RightAxis.Ticks.Color = clBlue
TopAxis.Grid.Visible = False
TopAxis.LabelsFont.Charset = DEFAULT_CHARSET
TopAxis.LabelsFont.Color = clGreen
TopAxis.LabelsFont.Height = -13
TopAxis.LabelsFont.Name = 'Arial'
TopAxis.LabelsFont.Style = []
TopAxis.Ticks.Color = clYellow
View3D = False
Align = alClient
BevelOuter = bvNone
BevelWidth = 6
Color = clSilver
TabOrder = 0
<-------> object FastLineSeries1: TFastLineSeries
HorizAxis = aTopAxis

```
    Height = 33
    TabOrder = 4
    Kind = bkClose
  end
  <-------> object Memo1: TMemo
    Left = 12
    Top = 144
    Width = 121
    Height = 181
    Lines.Strings = (
      'Use this demo to'
      'benchmark your'
      'video and system '
      'speed.'
      ''
      'Buffered display '
      'avoids flickering and '
      'can improve '
      'performance on modern '
      'hardware.'
      ''
      'FastLine Series are'
      'faster than regular'
      'Line Series.')
    TabOrder = 5
  end
 end
End
\\\\\\\\\\\\\\\\\\\\\\\\
24
<-------> object FeaturesForm:
TFeaturesForm
  Left = 186
  Top = 108
  BorderIcons = [biSystemMenu]
  BorderStyle = bsDialog
  Caption = 'TeeChart Features.
100% Native VCL / CLX'
  ClientHeight = 479
  ClientWidth = 609
  Color = clNavy
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clWhite
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
```

```
    Top = 8
    Width = 128
    Height = 33
    Caption = '&Speed Test !'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -16
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 0
    OnClick = Button1Click
  end
  <-------> object CheckBox1:
TCheckBox
    Left = 26
    Top = 52
    Width = 97
    Height = 17
    Caption = 'Buffered Display'
    TabOrder = 1
    OnClick = CheckBox1Click
  end
  <-------> object CheckBox2:
TCheckBox
    Left = 26
    Top = 76
    Width = 79
    Height = 17
    Caption = 'Draw &Axis'
    TabOrder = 2
    OnClick = CheckBox2Click
  end
  <-------> object CheckBox3:
TCheckBox
    Left = 26
    Top = 100
    Width = 75
    Height = 17
    Caption = '&Clip Points'
    TabOrder = 3
    OnClick = CheckBox3Click
  end
  <-------> object BitBtn3: TBitBtn
    Left = 28
    Top = 340
    Width = 89
```

Left = 104
Top = 76
Width = 109
Height = 16
HelpType = htKeyword
Caption = 'Animated Zoom'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label23: TLabel
Left = 104
Top = 111
Width = 146
Height = 16
HelpType = htKeyword
Caption = 'BackGround Bitmaps'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label24: TLabel
Left = 104
Top = 147
Width = 112
Height = 16
HelpType = htKeyword
Caption = 'Custom Drawing'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13

Position = poScreenCenter
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> object Label14: TLabel
Left = 16
Top = 457
Width = 300
Height = 13
HelpType = htKeyword
Caption = '© Copyright 1996-2001 by David Berneda. All Rights Reserved.'
Color = clNavy
Font.Charset = DEFAULT_CHARSET
Font.Color = clSilver
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentColor = False
ParentFont = False
end
<-------> object Panel2: TPanel
Left = 16
Top = 16
Width = 569
Height = 417
TabOrder = 20
<-------> object Label21: TLabel
Left = 104
Top = 40
Width = 140
Height = 16
HelpType = htKeyword
Caption = 'Custom Point Colors'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label22: TLabel

Caption = 'Overlayed Bars'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label28: TLabel
Left = 104
Top = 289
Width = 131
Height = 16
HelpType = htKeyword
Caption = 'Controlling Legend'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label29: TLabel
Left = 104
Top = 324
Width = 153
Height = 16
HelpType = htKeyword
Caption = 'Custom Axis Labelling'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True

Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label25: TLabel
Left = 104
Top = 182
Width = 108
Height = 16
HelpType = htKeyword
Caption = 'Custom Legend'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label26: TLabel
Left = 104
Top = 218
Width = 120
Height = 16
HelpType = htKeyword
Caption = 'Mini Small Charts'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label27: TLabel
Left = 104
Top = 253
Width = 109
Height = 16
HelpType = htKeyword

Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label33: TLabel
Left = 384
Top = 111
Width = 77
Height = 16
HelpType = htKeyword
Caption = 'Scroll Bars'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label34: TLabel
Left = 384
Top = 147
Width = 94
Height = 16
HelpType = htKeyword
Caption = 'Digital Charts'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label35: TLabel
Left = 384
Top = 182
Width = 74

end
<-------> object Label30: TLabel
Left = 104
Top = 360
Width = 145
Height = 16
HelpType = htKeyword
Caption = 'Custom Axis Drawing'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label31: TLabel
Left = 384
Top = 40
Width = 117
Height = 16
HelpType = htKeyword
Caption = 'Custom Scrolling'
Color = clSilver
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
Transparent = True
end
<-------> object Label32: TLabel
Left = 384
Top = 76
Width = 122
Height = 16
HelpType = htKeyword
Caption = 'TeeChart Shapes'
Color = clSilver
Font.Charset = DEFAULT_CHARSET

```
    Transparent = True
  end
  <-------> object Label38: TLabel
    Left = 384
    Top = 289
    Width = 114
    Height = 16
    HelpType = htKeyword
    Caption = 'Logarithmic Axis'
    Color = clSilver
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentColor = False
    ParentFont = False
    Transparent = True
  end
  <-------> object Label39: TLabel
    Left = 384
    Top = 324
    Width = 126
    Height = 16
    HelpType = htKeyword
    Caption = 'Chart Auto-Paging'
    Color = clSilver
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentColor = False
    ParentFont = False
    Transparent = True
  end
  <-------> object Label40: TLabel
    Left = 384
    Top = 360
    Width = 117
    Height = 16
    HelpType = htKeyword
    Caption = 'Metafiles *.WMF '
    Color = clSilver
    Font.Charset = DEFAULT_CHARSET
```

```
    Height = 16
    HelpType = htKeyword
    Caption = 'Cross Hair'
    Color = clSilver
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentColor = False
    ParentFont = False
    Transparent = True
  end
  <-------> object Label36: TLabel
    Left = 384
    Top = 218
    Width = 133
    Height = 16
    HelpType = htKeyword
    Caption = 'Keyboard Scrolling'
    Color = clSilver
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentColor = False
    ParentFont = False
    Transparent = True
  end
  <-------> object Label37: TLabel
    Left = 384
    Top = 253
    Width = 108
    Height = 16
    HelpType = htKeyword
    Caption = 'Custom Printing'
    Color = clSilver
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentColor = False
    ParentFont = False
```

```
  end
  <-------> object BitBtn9: TBitBtn
    Left = 40
    Top = 261
    Width = 65
    Height = 27
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 4
    OnClick = BitBtn9Click
    Glyph.Data = {
  end
  <-------> object BitBtn11: TBitBtn
    Left = 312
    Top = 332
    Width = 65
    Height = 27
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 5
    OnClick = BitBtn11Click
    Glyph.Data = {
    NumGlyphs = 2
  end
  <-------> object BitBtn10: TBitBtn
    Left = 312
    Top = 261
    Width = 65
    Height = 27
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 6
    OnClick = BitBtn10Click
```

```
      Font.Color = clBlack
      Font.Height = -13
      Font.Name = 'MS Sans Serif'
      Font.Style = [fsBold]
      ParentColor = False
      ParentFont = False
      Transparent = True
    end
    <-------> object Panel1: TPanel
      Left = 0
      Top = 0
      Width = 569
      Height = 17
      Caption = 'Choose a feature !'
      Color = clYellow
      Font.Charset =
DEFAULT_CHARSET
      Font.Color = clNavy
      Font.Height = -13
      Font.Name = 'MS Sans Serif'
      Font.Style = [fsBold]
      ParentFont = False
      TabOrder = 0
    end
  end
  <-------> object BitBtn5: TBitBtn
    Left = 40
    Top = 190
    Width = 65
    Height = 27
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 0
    OnClick = BitBtn5Click
    Glyph.Data = {

0003333333333333F777333333333333
0003333333333333F777333333333333
00

033333333FFF3F77733333333700073B
703333333F7773F77733333330777770
0B3
    NumGlyphs = 2
```

TabOrder = 9
OnClick = BitBtn12Click
Glyph.Data = {
end
<-------> object BitBtn13: TBitBtn
Left = 40
Top = 297
Width = 65
Height = 27
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 10
OnClick = BitBtn13Click
Glyph.Data = {
777770000000}
end
<-------> object BitBtn14: TBitBtn
Left = 312
Top = 155
Width = 65
Height = 27
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 11
OnClick = BitBtn14Click
Glyph.Data = {
end
<-------> object BitBtn15: TBitBtn
Left = 312
Top = 368
Width = 65
Height = 27
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []

Glyph.Data = {
NumGlyphs = 2
end
<-------> object BitBtn2: TBitBtn
Left = 312
Top = 48
Width = 65
Height = 27
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 7
OnClick = BitBtn2Click
Glyph.Data = {
end
<-------> object BitBtn8: TBitBtn
Left = 312
Top = 84
Width = 65
Height = 27
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 8
OnClick = BitBtn8Click
Glyph.Data = {
77777777777777770000}
end
<-------> object BitBtn12: TBitBtn
Left = 40
Top = 332
Width = 65
Height = 27
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False

Font.Style = []
ParentFont = False
TabOrder = 16
OnClick = BitBtn18Click
Glyph.Data = {
end
<-------> object BitBtn19: TBitBtn
Left = 312
Top = 190
Width = 65
Height = 27
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 17
OnClick = BitBtn19Click
Glyph.Data = {
NumGlyphs = 2
end
<-------> object BitBtn20: TBitBtn
Left = 312
Top = 226
Width = 65
Height = 27
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 18
OnClick = BitBtn20Click
Glyph.Data = {
NumGlyphs = 2
end
<-------> object BitBtn21: TBitBtn
Left = 312
Top = 297
Width = 65
Height = 27
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack

ParentFont = False
TabOrder = 13
OnClick = BitBtn15Click
Glyph.Data = {
end
<-------> object BitBtn16: TBitBtn
Left = 40
Top = 368
Width = 65
Height = 27
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 14
OnClick = BitBtn16Click
Glyph.Data = {
end
<-------> object BitBtn17: TBitBtn
Left = 40
Top = 48
Width = 65
Height = 27
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 15
OnClick = BitBtn17Click
Glyph.Data = {
8888888888888880000}
end
<-------> object BitBtn18: TBitBtn
Left = 312
Top = 119
Width = 65
Height = 27
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'

Height = 377
AnimatedZoom = True
AnimatedZoomSteps = 4
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
BackWall.Color = clSilver
Foot.Font.Charset = DEFAULT_CHARSET
Foot.Font.Color = clRed
Foot.Font.Height = -12
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsItalic]
Foot.Frame.Color = clScrollBar
Gradient.EndColor = clWhite
Gradient.StartColor = clYellow
Title.Alignment = taLeftJustify
Title.Brush.Color = 8454143
Title.Color = 8454143
Title.Font.Charset = DEFAULT_CHARSET
Title.Font.Color = clRed
Title.Font.Height = -19
Title.Font.Name = 'Arial'
Title.Font.Style = [fsBold, fsItalic]
Title.Frame.Color = clGreen
Title.Frame.Width = 4
Title.Text.Strings = (
'Zoom by clicking Left Mouse button'
'and dragging bottom right.')
BackColor = clSilver
BottomAxis.Grid.Color = clScrollBar
Chart3DPercent = 46
LeftAxis.Grid.Color = clScrollBar
Legend.Visible = False
RightAxis.Grid.Color = clScrollBar
TopAxis.Grid.Color = clScrollBar
View3D = False
Align = alClient
Color = clWhite
TabOrder = 0
<-------> object LineSeries1: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clRed

Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 19
OnClick = BitBtn21Click
Glyph.Data = {
<-------> object BitBtn22: TBitBtn
Left = 492
Top = 440
Width = 95
Height = 31
Caption = 'Close'
Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -11
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 21
Kind = bkClose
end
End
\\\\\\\\\\\\\\\\\\\\\\
25
<-------> object FormAnimatedZoom: TFormAnimatedZoom
Left = 197
Top = 113
Width = 619
Height = 411
ActiveControl = CheckBox1
Caption = 'TeeChart Animated Zoom Example'
Color = clBtnFace
ParentFont = True
OldCreateOrder = True
Position = poScreenCenter
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> object Chart1: TChart
Left = 141
Top = 0
Width = 470

XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 244
Top = 100
end
<-------> object LineSeries4: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clBlue
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 276
Top = 100
end
<-------> object LineSeries5: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clWhite
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 312

Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 172
Top = 100
end
<-------> object LineSeries2: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clGreen
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 208
Top = 100
end
<-------> object LineSeries3: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clYellow
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000

```
    Checked = True
    State = cbChecked
    TabOrder = 0
    OnClick = CheckBox1Click
  end
  <-------> object CheckBox2:
TCheckBox
    Left = 15
    Top = 12
    Width = 42
    Height = 17
    Caption = '3&D'
    TabOrder = 1
    OnClick = CheckBox2Click
  end
  <-------> object SpinEdit1:
TSpinEdit
    Left = 86
    Top = 64
    Width = 47
    Height = 22
    MaxValue = 1000
    MinValue = 1
    TabOrder = 2
    Value = 1
    OnChange = SpinEdit1Change
  end
  <-------> object BitBtn2: TBitBtn
    Left = 26
    Top = 140
    Width = 89
    Height = 33
    Caption = 'Zoom OUT'
    TabOrder = 3
    OnClick = BitBtn2Click
    Glyph.Data = {
  end
  <-------> object BitBtn3: TBitBtn
    Left = 26
    Top = 344
    Width = 89
    Height = 33
    TabOrder = 5
    OnClick = BitBtn2Click
    Kind = bkClose
  end
  <-------> object Memo1: TMemo
    Left = 12
```

```
    Top = 100
  end
  <-------> object PointSeries1:
TPointSeries
    Marks.ArrowLength = 8
    Marks.Visible = False
    SeriesColor = clGray
    Pointer.Brush.Color = 4259584
    Pointer.InflateMargins = True
    Pointer.Style = psDiamond
    Pointer.Visible = True
    XValues.DateTime = False
    XValues.Name = 'X'
    XValues.Multiplier =
1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier =
1.000000000000000000
    YValues.Order = loNone
    Left = 348
    Top = 100
  end
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 0
  Width = 141
  Height = 377
  Align = alLeft
  TabOrder = 1
  <-------> object Label1: TLabel
    Left = 14
    Top = 68
    Width = 60
    Height = 13
    HelpType = htKeyword
    Caption = 'Zoom &Steps:'
    FocusControl = SpinEdit1
  end
  <-------> object CheckBox1:
TCheckBox
    Left = 15
    Top = 32
    Width = 110
    Height = 17
    Caption = '&Animated Zoom !!!'
```

```
   LeftWall.Color = clWhite
   Title.Alignment = taLeftJustify
   Title.Brush.Color = clBlack
   Title.Color = clBlack
   Title.Font.Charset = DEFAULT_CHARSET
   Title.Font.Color = clYellow
   Title.Font.Height = -16
   Title.Font.Name = 'Arial'
   Title.Font.Style = [fsBold]
   Title.Frame.Color = clBlue
   Title.Frame.Width = 4
   Title.Frame.Visible = True
   Title.Text.Strings = (
     'Average,  High and Low.')
   BottomAxis.Axis.Color = clRed
   BottomAxis.DateTimeFormat = 'dd/MMM'
   BottomAxis.Grid.Color = clBlack
   BottomAxis.Grid.Style = psSolid
   BottomAxis.LabelsFont.Charset = DEFAULT_CHARSET
   BottomAxis.LabelsFont.Color = clPurple
   BottomAxis.LabelsFont.Height = -11
   BottomAxis.LabelsFont.Name = 'MS Sans Serif'
   BottomAxis.LabelsFont.Style = []
   BottomAxis.MinorTickCount = 0
   BottomAxis.TickLength = 9
   BottomAxis.Ticks.Color = clBlue
   Chart3DPercent = 45
   Frame.Width = 3
   LeftAxis.Axis.Color = clBlue
   LeftAxis.LabelsFont.Charset = DEFAULT_CHARSET
   LeftAxis.LabelsFont.Color = clRed
   LeftAxis.LabelsFont.Height = -13
   LeftAxis.LabelsFont.Name = 'Arial'
   LeftAxis.LabelsFont.Style = [fsBold]
   Legend.ShadowColor = clGray
   Legend.ShadowSize = 4
   RightAxis.Automatic = False
   RightAxis.AutomaticMaximum = False
```

```
     Top = 192
     Width = 121
     Height = 129
     Lines.Strings = (
       'Animated Zoom'
       'can be activated'
       'to better indentify'
       'zoomed areas.'
       ''
       'Try to zoom using'
       'the left mouse button,'
       'with and without'
       'Animated Zoom !')
     TabOrder = 6
   end
 end
End
\\\\\\\\\\\\\\
26
<-------> object HighLowForm: THighLowForm
  Left = 191
  Top = 110
  Width = 616
  Height = 440
  Caption = 'TeeChart Statistics Demo'
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 153
    Top = 0
    Width = 455
    Height = 406
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Width = 3
    Foot.Font.Charset = DEFAULT_CHARSET
    Foot.Font.Color = clRed
    Foot.Font.Height = -12
    Foot.Font.Name = 'Arial'
    Foot.Font.Style = [fsItalic]
```

```
      Pointer.Visible = True
      XValues.DateTime = True
      XValues.Name = 'X'
      XValues.Multiplier = 1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier = 1.000000000000000000
      YValues.Order = loNone
      YValues.ValueSource = 'Y'
      <-------> object AverageSeriesAverage: TAverageTeeFunction
      end
    end
    <-------> object HighSeries: TLineSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      DataSource = BarSeries1
      SeriesColor = clFuchsia
      Title = 'High'
      LinePen.Width = 2
      Pointer.Brush.Color = 8388863
      Pointer.InflateMargins = False
      Pointer.Pen.Color = clBlue
      Pointer.Pen.Style = psDot
      Pointer.Style = psRectangle
      Pointer.Visible = True
      XValues.DateTime = True
      XValues.Name = 'X'
      XValues.Multiplier = 1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier = 1.000000000000000000
      YValues.Order = loNone
      <-------> object HighSeriesHigh: THighTeeFunction
      end
    end
    <-------> object LowSeries: TLineSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
```

```
    RightAxis.AutomaticMinimum = False
    RightAxis.Grid.Visible = False
    RightAxis.LabelsFont.Charset = DEFAULT_CHARSET
    RightAxis.LabelsFont.Color = clBlack
    RightAxis.LabelsFont.Height = -19
    RightAxis.LabelsFont.Name = 'Arial'
    RightAxis.LabelsFont.Style = [fsBold, fsItalic]
    OnAfterDraw = Chart1AfterDraw
    Align = alClient
    BevelInner = bvRaised
    BevelOuter = bvLowered
    BorderWidth = 25
    Color = 16777088
    TabOrder = 0
    <-------> object BarSeries1: TBarSeries
      Marks.ArrowLength = 20
      Marks.Visible = True
      SeriesColor = clRed
      BarStyle = bsPyramid
      XValues.DateTime = True
      XValues.Name = 'X'
      XValues.Multiplier = 1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier = 1.000000000000000000
      YValues.Order = loNone
      Left = 47
      Top = 117
    end
    <-------> object AverageSeries: TLineSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      DataSource = BarSeries1
      SeriesColor = clGreen
      Title = 'Average'
      LinePen.Width = 2
      Pointer.InflateMargins = False
      Pointer.Style = psRectangle
```

```
    Left = 10
    Top = 12
    Width = 139
    Height = 25
    Caption = '&Animate !!!'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -21
    Font.Name = 'Arial'
    Font.Style = [fsBold, fsItalic]
    ParentFont = False
    TabOrder = 2
    OnClick = CheckBox1Click
  end
  <-------> object CB3D: TCheckBox
    Left = 12
    Top = 44
    Width = 49
    Height = 17
    Caption = '3D'
    TabOrder = 3
    OnClick = CB3DClick
  end
  <-------> object CheckBox2: TCheckBox
    Left = 12
    Top = 64
    Width = 101
    Height = 17
    Caption = 'Two Bar Series'
    TabOrder = 4
    OnClick = CheckBox2Click
  end
  <-------> object CheckBox3: TCheckBox
    Left = 12
    Top = 84
    Width = 97
    Height = 17
    Caption = 'Bar Visible'
    Checked = True
    State = cbChecked
    TabOrder = 5
    OnClick = CheckBox3Click
  end
  <-------> object Memo1: TMemo
```

```
    DataSource = BarSeries1
    SeriesColor = clBlue
    Title = 'Low'
    LinePen.Width = 2
    Pointer.InflateMargins = False
    Pointer.Style = psRectangle
    Pointer.Visible = True
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
    <-------> object LowSeriesLow: TLowTeeFunction
    end
   end
  end
  <-------> object Panel1: TPanel
   Left = 0
   Top = 0
   Width = 153
   Height = 406
   Align = alLeft
   TabOrder = 1
   <-------> object BitBtn1: TBitBtn
    Left = 25
    Top = 300
    Width = 89
    Height = 33
    Caption = '&Print'
    TabOrder = 0
    OnClick = BitBtn1Click
   end
   <-------> object BitBtn2: TBitBtn
    Left = 26
    Top = 344
    Width = 89
    Height = 33
    TabOrder = 1
    Kind = bkClose
   end
   <-------> object CheckBox1: TCheckBox
```

```
    BackWall.Brush.Style = bsClear
    Foot.Alignment = taRightJustify
    Foot.Font.Charset = DEFAULT_CHARSET
    Foot.Font.Color = clYellow
    Foot.Font.Height = -12
    Foot.Font.Name = 'Arial'
    Foot.Font.Style = [fsItalic]
    Foot.Frame.Color = clScrollBar
    Foot.Text.Strings = (

        'Left drag (Up/Down) to Zoom. Right drag to Scroll. Invert Zoom r' +
        'ectangle to reset.')
    Title.Font.Charset = DEFAULT_CHARSET
    Title.Font.Color = clWhite
    Title.Font.Height = -16
    Title.Font.Name = 'Arial'
    Title.Font.Style = [fsBold]
    Title.Frame.Color = clScrollBar
    Title.Text.Strings = (
      'This demo shows the Gantt chart Type.'
      'Each Point is a Horizontal Bar that defines a Date Period.'
      '')
    BottomAxis.LabelsAngle = 90
    BottomAxis.LabelsFont.Charset = DEFAULT_CHARSET
    BottomAxis.LabelsFont.Color = clWhite
    BottomAxis.LabelsFont.Height = -12
    BottomAxis.LabelsFont.Name = 'Arial'
    BottomAxis.LabelsFont.Style = []
    BottomAxis.LabelStyle = talValue
    BottomAxis.MinorTicks.Color = clWhite
    BottomAxis.TickInnerLength = 4
    BottomAxis.Ticks.Color = clLime
    BottomAxis.TicksInner.Color = 8388863
    BottomAxis.Title.Caption = 'Current Year'
    BottomAxis.Title.Font.Charset = DEFAULT_CHARSET
      Left = 12
      Top = 124
      Width = 121
      Height = 161
      Lines.Strings = (
        'Statistical functions'
        'can be applied to'
        'existing Series.'
        ''
        'This demo shows '
        'Average, High and Low'
        'functions on one or two'
        'normal Bar Series.')
      TabOrder = 6
    end
  end
  <-------> object Timer1: TTimer
    Enabled = False
    Interval = 100
    OnTimer = Timer1Timer
    Left = 224
    Top = 6
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
27
<-------> object GanttForm: TGanttForm
  Left = 147
  Top = 108
  Width = 686
  Height = 432
  Caption = 'TeeChart Gantt Series Demo'
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 137
    Top = 0
    Width = 541
    Height = 398
    AllowPanning = pmHorizontal
    BackWall.Brush.Color = clWhite
```

```
    Marks.Visible = True
    SeriesColor = clRed
    OnClick = GanttSeries1Click
    Pointer.Draw3D = False
    Pointer.HorizSize = 83
    Pointer.InflateMargins = True
    Pointer.Style = psRectangle
    Pointer.VertSize = 10
    Pointer.Visible = True
    XValues.DateTime = True
    XValues.Name = 'Start'
    XValues.Multiplier =
1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier =
1.000000000000000000
    YValues.Order = loNone
    ConnectingPen.Color = clRed
    ConnectingPen.Width = 2
    StartValues.DateTime = True
    StartValues.Name = 'Start'
    StartValues.Multiplier =
1.000000000000000000
    StartValues.Order = loAscending
    EndValues.DateTime = False
    EndValues.Name = 'End'
    EndValues.Multiplier =
1.000000000000000000
    EndValues.Order = loNone
    NextTask.DateTime = False
    NextTask.Name = 'NextTask'
    NextTask.Multiplier =
1.000000000000000000
    NextTask.Order = loNone
  end
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 0
  Width = 137
  Height = 398
  Align = alLeft
  TabOrder = 1
  <-------> object Label1: TLabel
    Left = 15
    Top = 11
```

```
   BottomAxis.Title.Font.Color =
clFuchsia
   BottomAxis.Title.Font.Height = -
19
   BottomAxis.Title.Font.Name =
'Arial'
   BottomAxis.Title.Font.Style =
[fsItalic]
   LeftAxis.Grid.Color = clLime
   LeftAxis.LabelsFont.Charset =
DEFAULT_CHARSET
   LeftAxis.LabelsFont.Color =
clAqua
   LeftAxis.LabelsFont.Height = -13
   LeftAxis.LabelsFont.Name =
'Arial'
   LeftAxis.LabelsFont.Style = []
   LeftAxis.LabelsSeparation = 3
   LeftAxis.LabelStyle = talText
   LeftAxis.Title.Font.Charset =
DEFAULT_CHARSET
   LeftAxis.Title.Font.Color =
clPurple
   LeftAxis.Title.Font.Height = -16
   LeftAxis.Title.Font.Name = 'Arial'
   LeftAxis.Title.Font.Style =
[fsBold, fsItalic]
   Legend.Visible = False
   View3D = False
   OnGetNextAxisLabel =
Chart1GetNextAxisLabel
   Align = alClient
   Color = 8421440
   TabOrder = 0
   OnMouseMove =
Chart1MouseMove
   <-------> object GanttSeries1:
TGanttSeries
     ColorEachPoint = True
     Marks.ArrowLength = 0
     Marks.BackColor = clWhite
     Marks.Clip = True
     Marks.Font.Charset =
DEFAULT_CHARSET
     Marks.Font.Color = clNavy
     Marks.Font.Height = -11
     Marks.Font.Name = 'Arial'
     Marks.Font.Style = []
     Marks.Frame.Color = clRed
```

```
      'Gantt Series '
      'are made of points'
      'with Start and End'
      'coordinates.'
      ''
      'This example shows'
      'how to get mouse '
      'position and how'
      'to customize bottom'
      'axis labels.')
     TabOrder = 1
   end
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\
28
<-------> object KeyboardForm: TKeyboardForm
  Left = 190
  Top = 116
  Width = 667
  Height = 389
  ActiveControl = Chart1
  Caption = 'TeeChart Keyboard Scrolling Example'
  Color = clBtnFace
  ParentFont = True
  KeyPreview = True
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  OnKeyDown = FormKeyDown
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 157
    Top = 0
    Width = 502
    Height = 355
    AnimatedZoom = True
    AnimatedZoomSteps = 3
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Color = clWhite
    BottomWall.Color = 8454016
    BottomWall.Size = 16
    Foot.Font.Charset = DEFAULT_CHARSET
```

```
      Width = 62
      Height = 24
      HelpType = htKeyword
      Caption = 'Label1'
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -20
      Font.Name = 'Arial'
      Font.Style = [fsBold]
      ParentFont = False
    end
    <-------> object Shape1: TShape
      Left = 24
      Top = 84
      Width = 77
      Height = 29
      HelpType = htKeyword
    end
    <-------> object Label2: TLabel
      Left = 15
      Top = 39
      Width = 62
      Height = 24
      HelpType = htKeyword
      Caption = 'Label1'
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clBlue
      Font.Height = -20
      Font.Name = 'Arial'
      Font.Style = [fsBold]
      ParentFont = False
    end
    <-------> object BitBtn3: TBitBtn
      Left = 24
      Top = 304
      Width = 89
      Height = 33
      TabOrder = 0
      Kind = bkClose
    end
    <-------> object Memo1: TMemo
      Left = 8
      Top = 128
      Width = 121
      Height = 161
      Lines.Strings = (
```

```
    Marks.ArrowLength = 8
    Marks.Visible = False
    SeriesColor = clRed
    Pointer.InflateMargins = True
    Pointer.Style = psRectangle
    Pointer.Visible = False
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier =
1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier =
1.000000000000000000
    YValues.Order = loNone
    Left = 199
    Top = 121
  end
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 0
  Width = 157
  Height = 355
  Align = alLeft
  TabOrder = 1
  <-------> object BitBtn1: TBitBtn
    Left = 37
    Top = 284
    Width = 89
    Height = 33
    TabOrder = 0
    Kind = bkClose
  end
  <-------> object InvertScroll:
TCheckBox
    Left = 21
    Top = 12
    Width = 128
    Height = 17
    Caption = '&Inverted Scrolling'
    TabOrder = 1
    OnClick = InvertScrollClick
  end
  <-------> object CheckLimits:
TCheckBox
    Left = 21
    Foot.Font.Color = clRed
    Foot.Font.Height = -12
    Foot.Font.Name = 'Arial'
    Foot.Font.Style = [fsItalic]
    Foot.Frame.Color = clScrollBar
    Foot.Text.Strings = (
     'The Form.OnKeyDown event
does the job. See source code. ')
    LeftWall.Size = 16
    Title.AdjustFrame = False
    Title.Brush.Color = 8453888
    Title.Color = 8453888
    Title.Font.Charset =
DEFAULT_CHARSET
    Title.Font.Color = clBlack
    Title.Font.Height = -12
    Title.Font.Name = 'Arial'
    Title.Font.Style = []
    Title.Frame.Color = 8388863
    Title.Frame.Width = 3
    Title.Frame.Visible = True
    Title.Text.Strings = (

      'This example shows how to
handle keyboard events to scroll
TChar' +
      't contents.'

      'Use the keyboard arrow keys
and  page up / page down keys to
scr' +
      'oll.'
     'Press SPACE to reset. Press
SHIFT and arrow keys to ZOOM.')
    BackColor = clWhite
    BottomAxis.Grid.Color =
clScrollBar
    Chart3DPercent = 20
    LeftAxis.Grid.Color = clScrollBar
    Legend.Visible = False
    RightAxis.Grid.Color =
clScrollBar
    TopAxis.Grid.Color = clScrollBar
    Align = alClient
    Color = 16777088
    TabOrder = 0
    TabStop = True
    <-------> object LineSeries1:
TLineSeries
```

```
 OnCreate = FormCreate
 PixelsPerInch = 96
 TextHeight = 16
 <-------> object Label14: TLabel
   Left = 17
   Top = 393
   Width = 300
   Height = 13
   HelpType = htKeyword
   Caption = '© Copyright 1996-2001 by David Berneda. All Rights Reserved.'
   Color = clNavy
   Font.Charset = DEFAULT_CHARSET
   Font.Color = clSilver
   Font.Height = -11
   Font.Name = 'MS Sans Serif'
   Font.Style = []
   ParentColor = False
   ParentFont = False
 end
 <-------> object GroupBox1: TGroupBox
   Left = 16
   Top = 8
   Width = 553
   Height = 233
   Color = clSilver
   Ctl3D = True
   Font.Charset = DEFAULT_CHARSET
   Font.Color = clBlack
   Font.Height = -13
   Font.Name = 'MS Sans Serif'
   Font.Style = [fsBold]
   ParentColor = False
   ParentCtl3D = False
   ParentFont = False
   TabOrder = 0
   <-------> object Label1: TLabel
     Left = 51
     Top = 36
     Width = 134
     Height = 16
     HelpType = htKeyword
     Alignment = taRightJustify
     Caption =
```

```
     Top = 32
     Width = 97
     Height = 17
     Caption = 'Check &Limits'
     TabOrder = 2
     OnClick = CheckLimitsClick
   end
   <-------> object Memo1: TMemo
     Left = 20
     Top = 76
     Width = 121
     Height = 161
     Lines.Strings = (
       'Scrolling and zooming'
       'can also be done using'
       'the keyboard.'
       ''
       'This example shows '
       'how to scroll and zoom'
       'a Chart using the Form'
       'OnKeyDown event.'
       '')
     TabOrder = 3
   end
  end
End
\\\\\\\\\
29
<-------> object SeriesForm: TSeriesForm
  Left = 190
  Top = 108
  BorderIcons = [biSystemMenu]
  BorderStyle = bsDialog
  Caption = 'TeeChart Component Library Demo. Series types.'
  ClientHeight = 414
  ClientWidth = 585
  Color = clNavy
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -13
  Font.Name = 'Arial'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  Scaled = False
```

```
    Left = 136
    Top = 155
    Width = 48
    Height = 16
    HelpType = htKeyword
    Alignment = taRightJustify
    Caption = 'Arrows'
    Color = clSilver
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentColor = False
    ParentFont = False
  end
  <-------> object Label5: TLabel
    Left = 112
    Top = 194
    Width = 69
    Height = 16
    HelpType = htKeyword
    Alignment = taRightJustify
    Caption = 'Statistical'
    Color = clSilver
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentColor = False
    ParentFont = False
  end
  <-------> object Label6: TLabel
    Left = 384
    Top = 135
    Width = 50
    Height = 16
    HelpType = htKeyword
    Caption = '&Bubble'
  end
  <-------> object Label7: TLabel
    Left = 384
    Top = 93
    Width = 37
    Height = 16
'&Line,Point,Area,Bar'
    Color = clSilver
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentColor = False
    ParentFont = False
  end
  <-------> object Label2: TLabel
    Left = 156
    Top = 76
    Width = 24
    Height = 16
    HelpType = htKeyword
    Alignment = taRightJustify
    Caption = '&Pie'
    Color = clSilver
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentColor = False
    ParentFont = False
  end
  <-------> object Label3: TLabel
    Left = 88
    Top = 115
    Width = 94
    Height = 16
    HelpType = htKeyword
    Alignment = taRightJustify
    Caption = 'Stacked Bars'
    Color = clSilver
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentColor = False
    ParentFont = False
  end
  <-------> object Label4: TLabel
```

```
    OnClick = BitBtn5Click
    Glyph.Data = {
  end
  <-------> object BitBtn6: TBitBtn
    Left = 309
    Top = 126
    Width = 57
    Height = 33
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -16
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 2
    OnClick = BitBtn6Click
    Glyph.Data = {
  end
  <-------> object BitBtn7: TBitBtn
    Left = 309
    Top = 84
    Width = 57
    Height = 33
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -16
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 3
    OnClick = BitBtn7Click
    Glyph.Data = {
  end
  <-------> object BStacked: TBitBtn
    Left = 200
    Top = 105
    Width = 57
    Height = 33
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -16
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 4
```

```
    HelpType = htKeyword
    Caption = '&Gantt'
  end
  <-------> object Label8: TLabel
    Left = 384
    Top = 52
    Width = 64
    Height = 16
    HelpType = htKeyword
    Caption = '&Fast Line'
  end
  <-------> object Label9: TLabel
    Left = 384
    Top = 176
    Width = 103
    Height = 16
    HelpType = htKeyword
    Caption = 'Stacked Areas'
  end
  <-------> object BitBtn4: TBitBtn
    Left = 200
    Top = 27
    Width = 57
    Height = 33
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -16
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 0
    OnClick = BitBtn4Click
    Glyph.Data = {
  end
  <-------> object BitBtn5: TBitBtn
    Left = 200
    Top = 66
    Width = 57
    Height = 33
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -16
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 1
```

```
    Font.Style = []
    ParentFont = False
    TabOrder = 7
    OnClick = BStackedAreasClick
    Glyph.Data = {
  end
  <-------> object BitBtn21: TBitBtn
    Left = 201
    Top = 183
    Width = 56
    Height = 35
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -16
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 8
    OnClick = BitBtn21Click
    Glyph.Data = {
    NumGlyphs = 2
  end
  <-------> object Panel1: TPanel
    Left = 0
    Top = 0
    Width = 553
    Height = 17
    Caption = 'Standard Chart
Types'
    Color = clYellow
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -13
    Font.Name = 'MS Sans Serif'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 9
  end
 end
 <-------> object GroupBox3:
TGroupBox
  Left = 16
  Top = 255
  Width = 553
  Height = 114
  Color = clSilver
```

```
    OnClick = BStackedClick
    Glyph.Data = {
  end
  <-------> object BFastLine:
TBitBtn
    Left = 309
    Top = 43
    Width = 57
    Height = 33
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -16
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 5
    OnClick = BFastLineClick
    Glyph.Data = {
  end
  <-------> object BArrowSeries:
TBitBtn
    Left = 200
    Top = 144
    Width = 57
    Height = 33
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -16
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 6
    OnClick = BArrowSeriesClick
    Glyph.Data = {
  end
  <-------> object BStackedAreas:
TBitBtn
    Left = 309
    Top = 167
    Width = 57
    Height = 33
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clNavy
    Font.Height = -16
    Font.Name = 'MS Sans Serif'
```

Font.Charset = DEFAULT_CHARSET
Font.Color = clNavy
Font.Height = -16
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 0
OnClick = BitBtn9Click
Glyph.Data = {
end
<-------> object BitBtn10: TBitBtn
Left = 200
Top = 68
Width = 57
Height = 33
Font.Charset = DEFAULT_CHARSET
Font.Color = clNavy
Font.Height = -16
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 1
OnClick = BitBtn10Click
Glyph.Data = {
end
<-------> object BitBtn11: TBitBtn
Left = 309
Top = 24
Width = 57
Height = 33
Font.Charset = DEFAULT_CHARSET
Font.Color = clNavy
Font.Height = -16
Font.Name = 'MS Sans Serif'
Font.Style = []
ParentFont = False
TabOrder = 2
OnClick = BitBtn11Click
Glyph.Data = {
end
<-------> object BitBtn14: TBitBtn
Left = 309
Top = 68
Width = 57
Height = 33

Font.Charset = DEFAULT_CHARSET
Font.Color = clBlack
Font.Height = -13
Font.Name = 'MS Sans Serif'
Font.Style = [fsBold]
ParentColor = False
ParentFont = False
TabOrder = 1
<-------> object Label10: TLabel
Left = 82
Top = 36
Width = 103
Height = 16
HelpType = htKeyword
Alignment = taRightJustify
Caption = 'P&ie from Table'
end
<-------> object Label11: TLabel
Left = 74
Top = 76
Width = 111
Height = 16
HelpType = htKeyword
Alignment = taRightJustify
Caption = '&SQL Query Bars'
end
<-------> object Label12: TLabel
Left = 384
Top = 34
Width = 100
Height = 16
HelpType = htKeyword
Caption = 'Lin&ked Tables'
end
<-------> object Label13: TLabel
Left = 384
Top = 74
Width = 132
Height = 16
HelpType = htKeyword
Caption = 'Horizontal DB Bars'
end
<-------> object BitBtn9: TBitBtn
Left = 200
Top = 24
Width = 57
Height = 33

```
TMiniForm
  Left = 187
  Top = 113
  Width = 192
  Height = 136
  Hint = 'Please Resize Me !!!'
  Caption = 'Mini-Tee-Charts.  Enjoy !!!'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText
  Font.Height = -14
  Font.Name = 'System'
  Font.Style = []
  OldCreateOrder = True
  ShowHint = True
  OnCreate = FormCreate
  OnResize = FormResize
  PixelsPerInch = 96
  TextHeight = 16
  <-------> object Panel1: TPanel
    Left = 0
    Top = 0
    Width = 97
    Height = 102
    Align = alLeft
    TabOrder = 0
    <-------> object Chart1: TChart
      Left = 1
      Top = 57
      Width = 95
      Height = 44
      BackWall.Brush.Color = clWhite
      BackWall.Brush.Style = bsClear
      BackWall.Pen.Visible = False
      Foot.Font.Charset = DEFAULT_CHARSET
      Foot.Font.Color = clRed
      Foot.Font.Height = -12
      Foot.Font.Name = 'Arial'
      Foot.Font.Style = [fsItalic]
      Foot.Frame.Color = clScrollBar
      MarginBottom = 0
      MarginLeft = 0
      MarginRight = 0
      MarginTop = 0
      Title.Frame.Color = clScrollBar
```

```
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clNavy
      Font.Height = -16
      Font.Name = 'MS Sans Serif'
      Font.Style = []
      ParentFont = False
      TabOrder = 3
      OnClick = BitBtn14Click
      Glyph.Data = {
    end
    <-------> object Panel2: TPanel
      Left = 0
      Top = 0
      Width = 553
      Height = 17
      Caption = 'Database Charts'
      Color = clYellow
      Font.Charset = DEFAULT_CHARSET
      Font.Color = clNavy
      Font.Height = -13
      Font.Name = 'MS Sans Serif'
      Font.Style = [fsBold]
      ParentFont = False
      TabOrder = 4
    end
  end
  <-------> object BitBtn1: TBitBtn
    Left = 472
    Top = 376
    Width = 97
    Height = 33
    Caption = '   &Close'
    Font.Charset = DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    ParentFont = False
    TabOrder = 2
    Kind = bkClose
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
30
<-------> object MiniForm:
```

XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 58
Top = 75
end
end
<-------> object Chart2: TChart
Left = 1
Top = 1
Width = 95
Height = 56
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
BackWall.Pen.Visible = False
Foot.Font.Charset = DEFAULT_CHARSET
Foot.Font.Color = clRed
Foot.Font.Height = -12
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsItalic]
Foot.Frame.Color = clScrollBar
MarginBottom = 0
MarginLeft = 0
MarginRight = 0
MarginTop = 0
Title.Frame.Color = clScrollBar
Title.Text.Strings = (
'TChart Component')
Title.Visible = False
BottomAxis.Visible = False
Chart3DPercent = 25
Frame.Visible = False
LeftAxis.Visible = False
Legend.Visible = False
RightAxis.Visible = False
TopAxis.Visible = False
View3D = False
View3DWalls = False
Align = alTop
TabOrder = 1
<-------> object BarSeries1: TBarSeries
ColorEachPoint = True
Marks.ArrowLength = 20

Title.Text.Strings = (
'TChart Component')
Title.Visible = False
BottomAxis.Visible = False
Chart3DPercent = 30
Frame.Visible = False
LeftAxis.Visible = False
Legend.Visible = False
RightAxis.Visible = False
TopAxis.Visible = False
View3DWalls = False
Align = alClient
TabOrder = 0
<-------> object LineSeries1: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clRed
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 15
Top = 74
end
<-------> object LineSeries2: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clGreen
AfterDrawValues = LineSeries2AfterDrawValues
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000

```
    Title.Text.Strings = (
      'TChart Component')
    Title.Visible = False
    AxisVisible = False
    BottomAxis.Visible = False
    ClipPoints = False
    Frame.Visible = False
    LeftAxis.Visible = False
    Legend.Visible = False
    RightAxis.Visible = False
    TopAxis.Visible = False
    View3DOptions.Elevation = 315
    View3DOptions.Orthogonal =
False
    View3DOptions.Perspective = 0
    View3DOptions.Rotation = 360
    View3DWalls = False
    Align = alTop
    TabOrder = 0
    <-------> object PieSeries1:
TPieSeries
      Marks.ArrowLength = 8
      Marks.Style = smsLabelPercent
      Marks.Visible = False
      SeriesColor = clRed
      Circled = True
      OtherSlice.Text = 'Other'
      PieValues.DateTime = False
      PieValues.Name = 'Y'
      PieValues.Multiplier =
1.000000000000000000
      PieValues.Order = loNone
      Left = 134
      Top = 13
    end
  end
  <-------> object Chart4: TChart
    Left = 1
    Top = 57
    Width = 85
    Height = 44
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    Foot.Font.Charset =
DEFAULT_CHARSET
    Foot.Font.Color = clRed
    Foot.Font.Height = -12
```

```
      Marks.Visible = False
      SeriesColor = clRed
      BarPen.Visible = False
      Dark3D = False
      XValues.DateTime = True
      XValues.Name = 'X'
      XValues.Multiplier =
1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier =
1.000000000000000000
      YValues.Order = loNone
      Left = 34
      Top = 13
    end
  end
end
<-------> object Panel2: TPanel
  Left = 97
  Top = 0
  Width = 87
  Height = 102
  Align = alClient
  TabOrder = 1
  <-------> object Chart3: TChart
    Left = 1
    Top = 1
    Width = 85
    Height = 56
    AllowPanning = pmNone
    AllowZoom = False
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    Foot.Font.Charset =
DEFAULT_CHARSET
    Foot.Font.Color = clRed
    Foot.Font.Height = -12
    Foot.Font.Name = 'Arial'
    Foot.Font.Style = [fsItalic]
    Foot.Frame.Color = clScrollBar
    MarginBottom = 0
    MarginLeft = 0
    MarginRight = 0
    MarginTop = 0
    Title.Frame.Color = clScrollBar
```

```
        SeriesColor = clGreen
        AfterDrawValues =
LineSeries3AfterDrawValues
        Pointer.InflateMargins = True
        Pointer.Style = psRectangle
        Pointer.Visible = False
        XValues.DateTime = True
        XValues.Name = 'X'
        XValues.Multiplier =
1.000000000000000000
        XValues.Order = loAscending
        YValues.DateTime = False
        YValues.Name = 'Y'
        YValues.Multiplier =
1.000000000000000000
        YValues.Order = loNone
        Left = 164
        Top = 79
      end
    end
  end
  <-------> object Timer1: TTimer
    Interval = 1
    OnTimer = Timer1Timer
    Left = 84
    Top = 43
  end
End
\\\\\\\\\\\\
31

<-------> object LinkedTablesForm:
TLinkedTablesForm
  Left = 185
  Top = 113
  Width = 604
  Height = 457
  Caption = 'TeeChart Linked Tables
Demo'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
```

```
      Foot.Font.Name = 'Arial'
      Foot.Font.Style = [fsItalic]
      Foot.Frame.Color = clScrollBar
      MarginBottom = 0
      MarginLeft = 0
      MarginRight = 0
      MarginTop = 0
      Title.Frame.Color = clScrollBar
      Title.Text.Strings = (
        'TChart Component')
      Title.Visible = False
      BottomAxis.Visible = False
      Frame.Visible = False
      LeftAxis.Visible = False
      Legend.Visible = False
      RightAxis.Visible = False
      TopAxis.Visible = False
      View3D = False
      View3DWalls = False
      Align = alClient
      TabOrder = 1
      <-------> object AreaSeries1:
TAreaSeries
        Marks.ArrowLength = 8
        Marks.Visible = False
        SeriesColor = clRed
        AreaLinesPen.Visible = False
        DrawArea = True
        Pointer.InflateMargins = True
        Pointer.Style = psRectangle
        Pointer.Visible = False
        XValues.DateTime = True
        XValues.Name = 'X'
        XValues.Multiplier =
1.000000000000000000
        XValues.Order = loAscending
        YValues.DateTime = False
        YValues.Name = 'Y'
        YValues.Multiplier =
1.000000000000000000
        YValues.Order = loNone
        Left = 111
        Top = 77
      end
      <-------> object LineSeries3:
TLineSeries
        Marks.ArrowLength = 8
        Marks.Visible = False
```

Legend.Font.Color = clYellow
Legend.Font.Height = -16
Legend.Font.Name = 'Arial'
Legend.Font.Style = [fsBold, fsItalic]
Legend.Frame.Color = clWhite
Legend.Frame.Width = 4
Legend.TextStyle = ltsRightValue
Legend.TopPos = 5
RightAxis.LabelsFont.Charset = DEFAULT_CHARSET
RightAxis.LabelsFont.Color = clRed
RightAxis.LabelsFont.Height = -15
RightAxis.LabelsFont.Name = 'Arial'
RightAxis.LabelsFont.Style = [fsItalic]
RightAxis.TickLength = 12
RightAxis.Ticks.Color = clWhite
RightAxis.Ticks.Width = 2
Align = alClient
TabOrder = 0
<-------> object AreaSeries1: TAreaSeries
ColorEachPoint = True
Marks.ArrowLength = 8
Marks.Visible = False
DataSource = Table2
SeriesColor = clRed
VertAxis = aRightAxis
XLabelsSource = 'SaleDate'
DrawArea = True
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = False
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
YValues.ValueSource = 'AmountPaid'

PixelsPerInch = 96
TextHeight = 13
<-------> object DBChart1: TDBChart
Left = 0
Top = 241
Width = 596
Height = 157
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
BackWall.Pen.Color = clAqua
BackWall.Pen.Style = psDashDot
Foot.Alignment = taRightJustify
Foot.Font.Charset = DEFAULT_CHARSET
Foot.Font.Color = clBlue
Foot.Font.Height = -15
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsBold]
Foot.Frame.Color = clScrollBar
Foot.Visible = False
Title.Font.Charset = DEFAULT_CHARSET
Title.Font.Color = clFuchsia
Title.Font.Height = -16
Title.Font.Name = 'Arial'
Title.Font.Style = [fsBold, fsItalic]
Title.Frame.Color = clScrollBar
Title.Text.Strings = (
'Linked Tables Chart Demo')
BottomAxis.LabelsAngle = 90
BottomAxis.LabelsFont.Charset = DEFAULT_CHARSET
BottomAxis.LabelsFont.Color = clGreen
BottomAxis.LabelsFont.Height = -13
BottomAxis.LabelsFont.Name = 'Arial'
BottomAxis.LabelsFont.Style = [fsBold]
BottomAxis.LabelStyle = talText
Chart3DPercent = 35
Frame.Color = clAqua
Frame.Style = psDashDot
Legend.Color = clBlue
Legend.ColorWidth = 13
Legend.Font.Charset = DEFAULT_CHARSET

```
    Left = 12
    Top = 11
    Width = 232
    Height = 26
    DataSource = DataSource1
    VisibleButtons = [nbFirst,
nbPrior, nbNext, nbLast]
    TabOrder = 0
  end
  <-------> object BitBtn1: TBitBtn
    Left = 286
    Top = 8
    Width = 95
    Height = 33
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -15
    Font.Name = 'Arial'
    Font.Style = []
    ParentFont = False
    TabOrder = 1
    Kind = bkClose
  end
 end
 <-------> object DBGrid2: TDBGrid
  Left = 0
  Top = 129
  Width = 596
  Height = 112
  Align = alTop
  DataSource = DataSource2
  TabOrder = 4
  TitleFont.Charset =
DEFAULT_CHARSET
  TitleFont.Color = clBlack
  TitleFont.Height = -11
  TitleFont.Name = 'MS Sans Serif'
  TitleFont.Style = []
 end
 <-------> object DataSource1:
TDataSource
  DataSet = Table1
  OnDataChange =
DataSource1DataChange
  Left = 339
  Top = 204
 end
```

```
    Left = 40
    Top = 258
   end
 end
 <-------> object DBGrid1: TDBGrid
  Left = 0
  Top = 48
  Width = 596
  Height = 81
  Align = alTop
  DataSource = DataSource1
  Options = [dgTitles, dgIndicator,
dgColumnResize, dgColLines,
dgRowLines, dgTabs, dgRowSelect,
dgAlwaysShowSelection,
dgConfirmDelete, dgCancelOnExit]
  TabOrder = 1
  TitleFont.Charset =
DEFAULT_CHARSET
  TitleFont.Color = clBlack
  TitleFont.Height = -11
  TitleFont.Name = 'MS Sans Serif'
  TitleFont.Style = []
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 398
  Width = 596
  Height = 25
  Align = alBottom
  Alignment = taLeftJustify
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clRed
  Font.Height = -13
  Font.Name = 'Arial'
  Font.Style = [fsItalic]
  ParentFont = False
  TabOrder = 2
 end
 <-------> object Panel2: TPanel
  Left = 0
  Top = 0
  Width = 596
  Height = 48
  Align = alTop
  TabOrder = 3
  <-------> object DBNavigator1:
TDBNavigator
```

```
    FieldName = 'CustNo'
    Required = True
    Visible = False
  end
  <-------> object Table2SaleDate:
TDateTimeField
    FieldName = 'SaleDate'
  end
  <-------> object
Table2AmountPaid:
TCurrencyField
    FieldName = 'AmountPaid'
  end
 end
 <-------> object DataSource2:
TDataSource
  DataSet = Table2
  Left = 336
  Top = 150
 end
End
\\\\\\\\\\\\\\\\\\\\
32
<-------> object PieForm: TPieForm
 Left = 178
 Top = 104
 Width = 642
 Height = 421
 Caption = 'TeeChart Pie Series
Demo'
 Color = clBtnFace
 ParentFont = True
 OldCreateOrder = True
 Position = poScreenCenter
 OnCreate = FormCreate
 PixelsPerInch = 96
 TextHeight = 13
 <-------> object Chart1: TChart
  Left = 137
  Top = 0
  Width = 497
  Height = 387
  AllowPanning = pmNone
  AllowZoom = False
  BackWall.Brush.Color = clWhite
  BackWall.Brush.Style = bsClear
  BackWall.Pen.Visible = False
  Foot.Alignment = taLeftJustify
```

```
 <-------> object Table1: TTable
  Active = True
  DatabaseName = 'DBDEMOS'
  TableName = 'CUSTOMER.DB'
  Left = 268
  Top = 203
  <-------> object Table1CustNo:
TFloatField
    DisplayWidth = 12
    FieldName = 'CustNo'
  end
  <-------> object Table1Company:
TStringField
    DisplayWidth = 29
    FieldName = 'Company'
    Size = 30
  end
  <-------> object Table1City:
TStringField
    DisplayWidth = 18
    FieldName = 'City'
    Size = 15
  end
  <-------> object Table1State:
TStringField
    DisplayWidth = 7
    FieldName = 'State'
  end
  <-------> object Table1Country:
TStringField
    DisplayWidth = 14
    FieldName = 'Country'
  end
 end
 <-------> object Table2: TTable
  Active = True
  DatabaseName = 'DBDEMOS'
  IndexFieldNames = 'CustNo'
  MasterFields = 'CustNo'
  MasterSource = DataSource1
  TableName = 'ORDERS.DB'
  Left = 268
  Top = 151
  <-------> object Table2OrderNo:
TFloatField
    FieldName = 'OrderNo'
  end
  <-------> object Table2CustNo:
TFloatField
```

```
[fsItalic]
  ClipPoints = False
  Frame.Visible = False
  LeftAxis.Grid.Color = clScrollBar
  LeftAxis.LabelsFont.Charset = DEFAULT_CHARSET
  LeftAxis.LabelsFont.Color = clWhite
  LeftAxis.LabelsFont.Height = -13
  LeftAxis.LabelsFont.Name = 'Arial'
  LeftAxis.LabelsFont.Style = []
  LeftAxis.Title.Caption = 'Stock Price'
  LeftAxis.Title.Font.Charset = DEFAULT_CHARSET
  LeftAxis.Title.Font.Color = clNavy
  LeftAxis.Title.Font.Height = -15
  LeftAxis.Title.Font.Name = 'Arial'
  LeftAxis.Title.Font.Style = [fsBold]
  RightAxis.Grid.Color = clScrollBar
  TopAxis.Grid.Color = clScrollBar
  View3DOptions.Elevation = 315
  View3DOptions.Orthogonal = False
  View3DOptions.Perspective = 0
  View3DOptions.Rotation = 360
  View3DWalls = False
  Align = alClient
  Color = clSilver
  TabOrder = 0
  OnMouseMove = Chart1MouseMove
  <-------> object PieSeries1: TPieSeries
    Cursor = 2020
    Marks.ArrowLength = 8
    Marks.Style = smsLabelPercent
    Marks.Visible = True
    SeriesColor = clRed
    OnClick = PieSeries1Click
    OtherSlice.Text = 'Other'
    PieValues.DateTime = False
    PieValues.Name = 'Y'
    PieValues.Multiplier = 1.000000000000000000
  Foot.Font.Charset = DEFAULT_CHARSET
  Foot.Font.Color = clRed
  Foot.Font.Height = -12
  Foot.Font.Name = 'Arial'
  Foot.Font.Style = [fsItalic]
  Foot.Frame.Color = clScrollBar
  Foot.Text.Strings = (
    '')
  Gradient.Direction = gdBottomTop
  Gradient.EndColor = 8454016
  Gradient.StartColor = 8454143
  Gradient.Visible = True
  Title.AdjustFrame = False
  Title.Alignment = taRightJustify
  Title.Font.Charset = DEFAULT_CHARSET
  Title.Font.Color = clBlue
  Title.Font.Height = -16
  Title.Font.Name = 'Arial'
  Title.Font.Style = [fsBold, fsItalic]
  Title.Frame.Color = clScrollBar
  Title.Text.Strings = (
    'Pie Chart Series Demo.')
  AxisVisible = False
  BottomAxis.Grid.Color = clScrollBar
  BottomAxis.LabelsAngle = 90
  BottomAxis.LabelsFont.Charset = DEFAULT_CHARSET
  BottomAxis.LabelsFont.Color = clTeal
  BottomAxis.LabelsFont.Height = -12
  BottomAxis.LabelsFont.Name = 'Arial'
  BottomAxis.LabelsFont.Style = []
  BottomAxis.Title.Caption = 'Stock Market Date'
  BottomAxis.Title.Font.Charset = DEFAULT_CHARSET
  BottomAxis.Title.Font.Color = clGreen
  BottomAxis.Title.Font.Height = -16
  BottomAxis.Title.Font.Name = 'Arial'
  BottomAxis.Title.Font.Style =
```

```
    Width = 109
    Height = 101
    Caption = 'Legend Position:'
    ItemIndex = 1
    Items.Strings = (
      'Left'
      'Right'
      'Top'
      'Bottom')
    TabOrder = 5
    OnClick = RadioGroup1Click
  end
  <-------> object RadioGroup2:
TRadioGroup
    Left = 12
    Top = 80
    Width = 109
    Height = 156
    Caption = 'Efects:'
    TabOrder = 8
  end
  <-------> object CheckBox2:
TCheckBox
    Left = 20
    Top = 104
    Width = 41
    Height = 17
    Caption = '&3D'
    Checked = True
    State = cbChecked
    TabOrder = 1
    OnClick = CheckBox2Click
  end
  <-------> object CheckBox3:
TCheckBox
    Left = 20
    Top = 130
    Width = 61
    Height = 17
    Caption = '&Circled'
    TabOrder = 2
    OnClick = CheckBox3Click
  end
  <-------> object CheckBox4:
TCheckBox
    Left = 20
    Top = 156
    Width = 97
```

```
    PieValues.Order = loNone
    Left = 78
    Top = 94
  end
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 0
  Width = 137
  Height = 387
  Align = alLeft
  TabOrder = 1
  <-------> object Shape1: TShape
    Left = 14
    Top = 40
    Width = 107
    Height = 33
    HelpType = htKeyword
  end
  <-------> object CheckBox1:
TCheckBox
    Left = 15
    Top = 6
    Width = 105
    Height = 29
    Caption = '&Animate !!!'
    Font.Charset =
DEFAULT_CHARSET
    Font.Color = clBlack
    Font.Height = -17
    Font.Name = 'Arial'
    Font.Style = [fsBold]
    ParentFont = False
    TabOrder = 0
    OnClick = CheckBox1Click
  end
  <-------> object BitBtn3: TBitBtn
    Left = 20
    Top = 352
    Width = 89
    Height = 29
    TabOrder = 3
    Kind = bkClose
  end
  <-------> object RadioGroup1:
TRadioGroup
    Left = 12
    Top = 244
```

```
Scroll Example'
  Color = clBtnFace
  Font.Charset =
DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 0
    Top = 53
    Width = 574
    Height = 323
    BackImageInside = True
    BackImageMode = pbmTile
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    Foot.Font.Charset =
DEFAULT_CHARSET
    Foot.Font.Color = clRed
    Foot.Font.Height = -12
    Foot.Font.Name = 'Arial'
    Foot.Font.Style = [fsItalic]
    Foot.Frame.Color = clScrollBar
    Title.Frame.Color = clScrollBar
    Title.Text.Strings = (
      'TChart Component')
    BottomAxis.Grid.Color =
clScrollBar
    LeftAxis.Grid.Color = clScrollBar
    RightAxis.Grid.Color =
clScrollBar
    TopAxis.Grid.Color = clScrollBar
    Align = alClient
    TabOrder = 0
    <-------> object LineSeries1:
TLineSeries
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clRed
      OnAfterAdd =
LineSeries1AfterAdd
      Pointer.InflateMargins = True
```

```
      Height = 17
      Caption = 'Use &Patterns'
      TabOrder = 4
      OnClick = CheckBox4Click
    end
    <-------> object CheckBox5:
TCheckBox
      Left = 20
      Top = 182
      Width = 61
      Height = 17
      Caption = '&Marks'
      Checked = True
      State = cbChecked
      TabOrder = 6
      OnClick = CheckBox5Click
    end
    <-------> object CheckBox6:
TCheckBox
      Left = 20
      Top = 208
      Width = 65
      Height = 17
      Caption = '&Gradient'
      Checked = True
      State = cbChecked
      TabOrder = 7
      OnClick = CheckBox6Click
    end
  end
  <-------> object Timer1: TTimer
    Enabled = False
    Interval = 1
    OnTimer = Timer1Timer
    Left = 114
    Top = 294
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
33
<-------> object ScrollForm:
TScrollForm
  Left = 191
  Top = 113
  Width = 582
  Height = 410
  ActiveControl = Button1
  Caption = 'TeeChart - Automatic
```

```
    TabOrder = 2
    OnClick = CBVerticalClick
  end
 end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
34
<-------> object LegendForm: TLegendForm
 Left = 186
 Top = 115
 Width = 543
 Height = 385
 Caption = 'TeeChart Customized Legend Example'
 Color = clBtnFace
 ParentFont = True
 OldCreateOrder = True
 Position = poScreenCenter
 OnCreate = FormCreate
 PixelsPerInch = 96
 TextHeight = 13
 <-------> object PaintBox1: TPaintBox
  Left = 0
  Top = 0
  Width = 535
  Height = 52
  HelpType = htKeyword
  Align = alTop
 end
 <-------> object Label1: TLabel
  Left = 8
  Top = 4
  Width = 123
  Height = 13
  HelpType = htKeyword
  Caption = 'The Chart Legend is here:'
 end
 <-------> object Chart1: TChart
  Left = 153
  Top = 52
  Width = 382
  Height = 299
  BackWall.Brush.Color = clWhite
  BackWall.Brush.Style = bsClear
  Foot.Frame.Color = clScrollBar
```

```
    Pointer.Style = psRectangle
    Pointer.Visible = False
    XValues.DateTime = True
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
    Left = 119
    Top = 94
  end
 end
 <-------> object Panel1: TPanel
  Left = 0
  Top = 0
  Width = 574
  Height = 53
  Align = alTop
  TabOrder = 1
  <-------> object Button1: TButton
    Left = 191
    Top = 9
    Width = 250
    Height = 33
    Caption = 'Click here to add a new point and scroll !!!'
    TabOrder = 0
    OnClick = Button1Click
  end
  <-------> object BitBtn3: TBitBtn
    Left = 8
    Top = 12
    Width = 89
    Height = 33
    TabOrder = 1
    Kind = bkClose
  end
  <-------> object CBVertical: TCheckBox
    Left = 110
    Top = 18
    Width = 71
    Height = 17
    Caption = '&Vertical'
```

Top = 108
end
end
<-------> object Panel1: TPanel
Left = 0
Top = 52
Width = 153
Height = 299
Align = alLeft
TabOrder = 1
<-------> object BitBtn3: TBitBtn
Left = 30
Top = 224
Width = 89
Height = 33
TabOrder = 0
Kind = bkClose
end
<-------> object Memo1: TMemo
Left = 16
Top = 24
Width = 121
Height = 161
Lines.Strings = (
'Chart Legends can'
'be customized using'
'any Delphi component.'
''
'Here a TPaintBox is '
'used to display'
'Series Titles.'
'')
TabOrder = 1
end
end
End
\\\ \\\\\
36
<-------> object OverBarForm: TOverBarForm
Left = 187
Top = 107
Width = 598
Height = 397
Caption = 'TeeChart Overlayed Bars Example'
Color = clBtnFace

Title.Frame.Color = clScrollBar
Title.Text.Strings = (
'TChart Component')
Title.Visible = False
Legend.Visible = False
Align = alClient
TabOrder = 0
<-------> object LineSeries1: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clRed
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 63
Top = 110
end
<-------> object LineSeries2: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clGreen
AfterDrawValues = LineSeries2AfterDrawValues
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 129

```
  <-------> object BarSeries3: TBarSeries
    Cursor = crUpArrow
    Marks.ArrowLength = 20
    Marks.Visible = False
    SeriesColor = clBlue
    OnClick = BarSeries3Click
    BarWidthPercent = 80
    MultiBar = mbNone
    XValues.DateTime = False
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
    Left = 24
    Top = 36
  end
  <-------> object BarSeries2: TBarSeries
    Cursor = crUpArrow
    Marks.ArrowLength = 20
    Marks.Visible = False
    SeriesColor = clRed
    OnClick = BarSeries2Click
    BarWidthPercent = 60
    MultiBar = mbNone
    XValues.DateTime = False
    XValues.Name = 'X'
    XValues.Multiplier = 1.000000000000000000
    XValues.Order = loAscending
    YValues.DateTime = False
    YValues.Name = 'Y'
    YValues.Multiplier = 1.000000000000000000
    YValues.Order = loNone
    Left = 199
    Top = 122
  end
  <-------> object BarSeries1: TBarSeries
    Cursor = crUpArrow
    Marks.ArrowLength = 20
    Marks.Visible = False
```

```
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 149
    Top = 0
    Width = 441
    Height = 363
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    Foot.Font.Charset = DEFAULT_CHARSET
    Foot.Font.Color = clRed
    Foot.Font.Height = -12
    Foot.Font.Name = 'Arial'
    Foot.Font.Style = [fsItalic]
    Foot.Frame.Color = clScrollBar
    MarginTop = 2
    Title.Brush.Color = 8421440
    Title.Color = 8421440
    Title.Font.Charset = DEFAULT_CHARSET
    Title.Font.Color = clAqua
    Title.Font.Height = -19
    Title.Font.Name = 'Arial'
    Title.Font.Style = [fsBold, fsItalic]
    Title.Frame.Color = clBlue
    Title.Frame.Width = 4
    Title.Text.Strings = (
      'Overlayed Bar Series Example')
    BottomAxis.Grid.Color = clScrollBar
    LeftAxis.Grid.Color = clScrollBar
    Legend.Alignment = laLeft
    RightAxis.Grid.Color = clScrollBar
    TopAxis.Grid.Color = clScrollBar
    View3D = False
    Align = alClient
    TabOrder = 0
```

```
      'OffsetPercent and'
      'BarWidthPercent'
      'properties of '
      'TBarSeries to create'
      'overlayed bars.'
      ''
      'Bars can be displayed'
      'using pattern brushes.'
      '')
    TabOrder = 0
  end
  <-------> object BitBtn3: TBitBtn
    Left = 28
    Top = 316
    Width = 89
    Height = 33
    Caption = 'Close'
    TabOrder = 1
    Kind = bkClose
  end
  <-------> object SpinEdit1: TSpinEdit
    Left = 75
    Top = 18
    Width = 50
    Height = 22
    Increment = 5
    MaxValue = 100
    MinValue = 1
    TabOrder = 2
    Value = 60
    OnChange = SpinEdit1Change
  end
  <-------> object SpinEdit2: TSpinEdit
    Left = 75
    Top = 46
    Width = 50
    Height = 22
    Increment = 5
    MaxValue = 100
    MinValue = -100
    TabOrder = 3
    Value = 0
    OnChange = SpinEdit2Change
  end
  <-------> object CBPatterns: TCheckBox
```

```
      SeriesColor = clLime
      OnClick = BarSeries1Click
      BarWidthPercent = 40
      MultiBar = mbNone
      XValues.DateTime = False
      XValues.Name = 'X'
      XValues.Multiplier = 1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier = 1.000000000000000000
      YValues.Order = loNone
      Left = 127
      Top = 46
    end
  end
  <-------> object Panel1: TPanel
    Left = 0
    Top = 0
    Width = 149
    Height = 363
    Align = alLeft
    TabOrder = 1
    <-------> object Label1: TLabel
      Left = 15
      Top = 22
      Width = 48
      Height = 13
      HelpType = htKeyword
      Caption = 'Overlay %:'
    end
    <-------> object Label2: TLabel
      Left = 23
      Top = 46
      Width = 40
      Height = 13
      HelpType = htKeyword
      Caption = 'Offset %:'
    end
    <-------> object Memo1: TMemo
      Left = 16
      Top = 136
      Width = 121
      Height = 161
      Lines.Strings = (
        'This demo uses the'
```

```
   Foot.Font.Name = 'Arial'
   Foot.Font.Style = [fsItalic]
   Foot.Frame.Color = clScrollBar
   Title.Frame.Color = clScrollBar
   Title.Text.Strings = (
     'TChart Component')
   BottomAxis.Grid.Color = clScrollBar
   Chart3DPercent = 60
   LeftAxis.Grid.Color = clScrollBar
   LeftAxis.Grid.Width = 0
   RightAxis.Grid.Color = clScrollBar
   TopAxis.Grid.Color = clScrollBar
   TabOrder = 0
   <-------> object LineSeries1: TLineSeries
     Marks.ArrowLength = 8
     Marks.Visible = False
     SeriesColor = clRed
     Pointer.InflateMargins = True
     Pointer.Style = psRectangle
     Pointer.Visible = False
     XValues.DateTime = True
     XValues.Name = 'X'
     XValues.Multiplier = 1.000000000000000000
     XValues.Order = loAscending
     YValues.DateTime = False
     YValues.Name = 'Y'
     YValues.Multiplier = 1.000000000000000000
     YValues.Order = loNone
     Left = 147
     Top = 118
   end
 end
 <-------> object BitBtn1: TBitBtn
   Left = 38
   Top = 310
   Width = 119
   Height = 33
   Caption = 'PRINT ALL !!'
   TabOrder = 1
   OnClick = BitBtn1Click
   Glyph.Data = {

76010000424D7601000000000000760
000002800000020000000100000001
```

```
     Left = 19
     Top = 82
     Width = 90
     Height = 17
     Caption = 'Use Patterns'
     TabOrder = 4
     OnClick = CBPatternsClick
   end
 end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\
37
<-------> object PrintForm: TPrintForm
  Left = 201
  Top = 108
  BorderStyle = bsDialog
  Caption = 'TeeChart Printing Demo'
  ClientHeight = 349
  ClientWidth = 299
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  Scaled = False
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Label1: TLabel
    Left = 9
    Top = 4
    Width = 72
    Height = 13
    HelpType = htKeyword
    Caption = 'Enter text here:'
  end
  <-------> object Chart1: TChart
    Left = 5
    Top = 55
    Width = 284
    Height = 250
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    Foot.Font.Charset = DEFAULT_CHARSET
    Foot.Font.Color = clRed
    Foot.Font.Height = -12
```

```
    Width = 280
    Height = 21
    TabOrder = 2
    Text = 'Hello World !'
  end
  <-------> object BitBtn2: TBitBtn
    Left = 201
    Top = 311
    Width = 89
    Height = 33
    TabOrder = 3
    Kind = bkClose
  end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\


38
\\\\\\\\\\\\\\\\\\\\\\\\\\\\
<-------> object LegendXYForm:
TLegendXYForm
  Left = 200
  Top = 110
  Width = 569
  Height = 370
  Caption = 'TeeChart Example.
Customizing Legend Size and
Position'
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
  <-------> object Chart1: TChart
    Left = 141
    Top = 0
    Width = 420
    Height = 336
    AllowPanning = pmNone
    AllowZoom = False
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    BackWall.Pen.Visible = False
    Foot.Frame.Color = clScrollBar
    Title.Frame.Color = clScrollBar
    Title.Text.Strings = (
```

```
00

0400000000000000010000120B0000120
B000010000000000000000000000000000
00

800000800000008080008000000000800
08000808000007F7F7F00BFBFBF00
0000

FF0000FF000000FFFF00FF000000F
F00FF00FFFF0000FFFFFF0030000
0000000

000337777777777777730888888888
888807F333333333333370888888888
88

88807FFFFFFFFFFFFF700000000
000000007777777777777770F8F8F
8F8F8F

8F807F3333333333333F708F8F8F8F8
F8F9F07F33333333337370F8F8F8
F8F8F

8F807FFFFFFFFFFFFFF700000000
000000007777777777777773330FF
FFFFFF

03333337F3FFFF3F7F333330F0000
F0F03333337F77773737F333330FFF
FFFFF

03333337F3FF3FFF7F333330F00F0
00003333337F773777773333330FFF
F0FF0

33333337F3F37F3733333330F08F0F
0333333337F7337F7333333330FFFF
0033

33333337FFFF7733333333330000003
33333333377777773333333}
    NumGlyphs = 2
  end
  <-------> object Edit1: TEdit
    Left = 9
    Top = 23
```

```
      'Default'
      'Customized')
    TabOrder = 0
    OnClick = RadioGroup1Click
  end
  <-------> object BitBtn3: TBitBtn
    Left = 25
    Top = 272
    Width = 89
    Height = 33
    TabOrder = 1
    Kind = bkClose
  end
  <-------> object Memo1: TMemo
    Left = 8
    Top = 88
    Width = 121
    Height = 161
    Lines.Strings = (
      'Using the '
      'OnGetLegendPos and'
      'OnGetLegendRect'
      'events is possible'
      'to customize the Chart'
      'Legend position and'
      'Legend items positions.')
    TabOrder = 2
  end
 end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
39
<-------> object ScrollBarForm: TScrollBarForm
  Left = 196
  Top = 108
  Width = 565
  Height = 331
  Caption = 'TeeChart Scroll Bars Example'
  Color = clBtnFace
  ParentFont = True
  OldCreateOrder = True
  Position = poScreenCenter
  Scaled = False
  OnCreate = FormCreate
  PixelsPerInch = 96
  TextHeight = 13
```

```
      'TChart Component')
    OnGetLegendPos = Chart1GetLegendPos
    OnGetLegendRect = Chart1GetLegendRect
    AxisVisible = False
    ClipPoints = False
    Frame.Visible = False
    View3DOptions.Elevation = 315
    View3DOptions.Orthogonal = False
    View3DOptions.Perspective = 0
    View3DOptions.Rotation = 360
    View3DWalls = False
    Align = alClient
    TabOrder = 0
    <-------> object PieSeries1: TPieSeries
      Marks.ArrowLength = 8
      Marks.Style = smsPercent
      Marks.Visible = True
      SeriesColor = clRed
      OtherSlice.Text = 'Other'
      PieValues.DateTime = False
      PieValues.Name = 'Y'
      PieValues.Multiplier = 1.000000000000000000
      PieValues.Order = loNone
      Left = 287
      Top = 2
    end
  end
  <-------> object Panel1: TPanel
    Left = 0
    Top = 0
    Width = 141
    Height = 336
    Align = alLeft
    TabOrder = 1
    <-------> object RadioGroup1: TRadioGroup
      Left = 14
      Top = 12
      Width = 103
      Height = 61
      Caption = 'Legend Drawing'
      ItemIndex = 1
      Items.Strings = (
```

Top = 266
Width = 523
Height = 19
PageSize = 0
TabOrder = 1
OnChange = ScrollBar1Change
end
<-------> object LineSeries1: TLineSeries
Marks.ArrowLength = 8
Marks.Visible = False
SeriesColor = clRed
Pointer.InflateMargins = True
Pointer.Style = psRectangle
Pointer.Visible = False
XValues.DateTime = True
XValues.Name = 'X'
XValues.Multiplier = 1.000000000000000000
XValues.Order = loAscending
YValues.DateTime = False
YValues.Name = 'Y'
YValues.Multiplier = 1.000000000000000000
YValues.Order = loNone
Left = 199
Top = 121
end
end
End
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
40
<-------> object LogLabelsForm: TLogLabelsForm
Left = 199
Top = 108
Width = 631
Height = 411
Caption = 'Logarithmic Labels example'
Color = clBtnFace
ParentFont = True
OldCreateOrder = True
Position = poScreenCenter
OnCreate = FormCreate
PixelsPerInch = 96
TextHeight = 13
<-------> object Chart1: TChart

<-------> object Chart1: TChart
Left = 0
Top = 0
Width = 557
Height = 297
BackWall.Brush.Color = clWhite
BackWall.Brush.Style = bsClear
Foot.Font.Charset = DEFAULT_CHARSET
Foot.Font.Color = clRed
Foot.Font.Height = -12
Foot.Font.Name = 'Arial'
Foot.Font.Style = [fsItalic]
Foot.Frame.Color = clScrollBar
MarginBottom = 18
MarginRight = 18
Title.Frame.Color = clScrollBar
Title.Text.Strings = (
'TChart Component')
OnScroll = Chart1Scroll
OnUndoZoom = Chart1UndoZoom
OnZoom = Chart1Zoom
BottomAxis.Grid.Color = clScrollBar
LeftAxis.Grid.Color = clScrollBar
Legend.Visible = False
RightAxis.Grid.Color = clScrollBar
TopAxis.Grid.Color = clScrollBar
View3D = False
Align = alClient
TabOrder = 0
OnResize = Chart1Resize
<-------> object ScrollBar2: TScrollBar
Left = 517
Top = 10
Width = 19
Height = 255
Kind = sbVertical
PageSize = 0
TabOrder = 0
OnChange = ScrollBar2Change
end
<-------> object ScrollBar1: TScrollBar
Left = 13

```
    Left = 153
    Top = 0
    Width = 470
    Height = 377
    BackWall.Brush.Color = clWhite
    BackWall.Brush.Style = bsClear
    Gradient.Direction = gdLeftRight
    Gradient.Visible = True
    Title.Text.Strings = (
      'TChart')
    BottomAxis.Logarithmic = True
    LeftAxis.Logarithmic = True
    View3D = False
    Align = alClient
    TabOrder = 0
    <-------> object Series1:
TFastLineSeries
      ColorEachPoint = True
      Marks.ArrowLength = 8
      Marks.Visible = False
      SeriesColor = clRed
      LinePen.Color = clRed
      LinePen.Width = 2
      XValues.DateTime = False
      XValues.Name = 'X'
      XValues.Multiplier =
1.000000000000000000
      XValues.Order = loAscending
      YValues.DateTime = False
      YValues.Name = 'Y'
      YValues.Multiplier =
1.000000000000000000
      YValues.Order = loNone
    end
  end
  <-------> object Panel1: TPanel
    Left = 0
    Top = 0
    Width = 153
    Height = 377
    Align = alLeft
    TabOrder = 1
    <-------> object BitBtn2: TBitBtn
      Left = 26
      Top = 260
      Width = 89
      Height = 33
      Caption = 'Close'
      TabOrder = 0
      Kind = bkClose
    end
    <-------> object Memo1: TMemo
      Left = 12
      Top = 72
      Width = 121
      Height = 149
      Lines.Strings = (
        'Logarithmic Axis'
        'can draw labels in'
        'decades using the'
        'OnGetNextAxisLabel'
        'and OnGetAxisLabel'
        'events.'
        ''
        '')
      TabOrder = 1
    end
  end
End
```

الملف التنفيذي لبرنامج

(TeeChart)

إعداد

علاء الدين اللباد

هاتف ٠٩٤٤٥٧٥٣٧١

برنامج يتعلق بالرسوم البيانية والإحصائية يحتوي العديد من الميزات الرائعة في برنامج دلفي ٧ وهو موجود كمثال باسم ضمن البرنامج نفسه ويمكن الاستفادة من جميع الميزات الموجودة فيه قام بتحليل وشرح البرنامج علاء الدين اللباد ٠٩٤٤٥٧٥٣٧١

الملف التنفيذي للبرنامج ويحتوي على ٥٤٢ إجراء

ProcedureS 542

وجميعها معلمة بالسهم

<----->

```
{*******************************
**************}
{ TeeChart Delphi Component
```

```
Library 3.0      }
{ Demo                          }
{ Copyright (c) 1995-2001 by David Berneda   }
{ All rights reserved                     }
{*********************************************}
unit teemain;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, ExtCtrls, TeEngine, Series, TeeProcs, Chart, StdCtrls,
  TeeFunci;

type
  TTeeMainForm = class(TForm)
    Panel1: TPanel;
    Button1: TButton;
    Image1: TImage;
    Timer1: TTimer;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Button6: TButton;
    Chart2: TChart;
    FastLineSeries1: TFastLineSeries;
    FastLineSeries2: TFastLineSeries;
    Timer2: TTimer;
    Label11: TLabel;
    Label12: TLabel;
    Panel2: TPanel;
    Chart1: TChart;
    Series1: TPieSeries;
    Chart7: TChart;
    PieSeries6: TPieSeries;
    Chart8: TChart;
    PieSeries7: TPieSeries;
    Chart9: TChart;
    PieSeries8: TPieSeries;
    Chart10: TChart;
    PieSeries9: TPieSeries;
    Label13: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    Label2: TLabel;
    Panel3: TPanel;
    Label1: TLabel;
    Label9: TLabel;
    Label3: TLabel;
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure Timer1Timer(Sender: TObject);
    <-------> procedure Button1Click(Sender: TObject);
    <-------> procedure Timer2Timer(Sender: TObject);
    <-------> procedure Button2Click(Sender: TObject);
    <-------> procedure Chart2MouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    <-------> procedure Chart2MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    <-------> procedure Label10Click(Sender: TObject);
    <-------> procedure Button3Click(Sender: TObject);
    <-------> procedure Button4Click(Sender: TObject);
    <-------> procedure Button6Click(Sender: TObject);
    <-------> procedure Label3Click(Sender: TObject);
  private
    { Private declarations }
    DeltaZoom:Integer;
  public
    { Public declarations }
    <-------> procedure ShowForm(FormClass: TFormClass);
  end;

var
  TeeMainForm: TTeeMainForm;
```

```
  PieSeries7.Rotate(5);
  PieSeries9.Rotate(5);
  Series1.Rotate(355);
  PieSeries8.Rotate(355);
  With Chart9.View3DOptions do
  begin
    Zoom:=Zoom-DeltaZoom;
    if (Zoom<60) or (Zoom>110) then DeltaZoom:=-DeltaZoom;
  end;
  With PieSeries8 do
  begin
    tmp:=Random(Count);
    if ExplodedSlice[tmp]>20 then
      ExplodedSlice[tmp]:=ExplodedSlice[tmp]-1
    else
      ExplodedSlice[tmp]:=ExplodedSlice[tmp]+1;
  end;
end;

<-------> procedure TTeeMainForm.Button1Click(Sender: TObject);
begin
  Close;
end;

<-------> procedure TTeeMainForm.Timer2Timer(Sender: TObject);
var tmpX:Double;
begin
  With FastLineSeries1 do
  begin
    tmpX:=XValues[1]-XValues[0];
    Delete(0);
    AddXY(XValues.Last+tmpX,Random(100),'',clTeeColor);
  end;
  With FastLineSeries2 do
  begin
    tmpX:=XValues[1]-XValues[0];
    Delete(0);
```

```

implementation

{$R *.dfm}

Uses TeeAbout, TeeBasic, Features, UDemoCha, Specs, ShellAPI;

<-------> procedure TTeeMainForm.ShowForm(FormClass: TFormClass);
begin
  Timer1.Enabled:=False;
  Timer2.Enabled:=False;
  With FormClass.Create(Self) do
  try
    ShowModal;
  finally
    Free;
  end;
  Timer2.Enabled:=True;
  Timer1.Enabled:=True;
end;

<-------> procedure TTeeMainForm.FormCreate(Sender: TObject);
begin
  TeeEraseBack:=False;
  PieSeries6.FillSampleValues(5);
  PieSeries9.FillSampleValues(5);
  PieSeries7.FillSampleValues(5);
  Series1.FillSampleValues(5);
  PieSeries8.CheckDataSource;
  FastLineSeries1.FillSampleValues(20);
  FastLineSeries2.FillSampleValues(20);
  DeltaZoom:=5;
end;

<-------> procedure TTeeMainForm.Timer1Timer(Sender: TObject);
var tmp:Integer;
begin
  PieSeries6.Rotate(5);
```

```
  ShowForm(TChartSpecs);
end;

<-------> procedure
TTeeMainForm.Button6Click(Sende
r: TObject);
begin
  ShowForm(TSeriesForm);
end;

<-------> procedure
TTeeMainForm.Label3Click(Sender
: TObject);
Var St:Array[0..255] of char;
begin
  Timer1.Enabled:=False;
  Timer2.Enabled:=False;

ShellExecute(0,'open',StrPCopy(St,'
http://'+Label3.Caption+

'/products/teechart/delphi6'),nil,nil,S
W_SHOW);
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

program Teedemo;

uses
  Forms,
  teemain in 'teemain.PAS'
{TeeMainForm},
  Teebasic in 'teebasic.PAS'
{DemoForm},
  Features in 'features.pas'
{FeaturesForm},
  Udemocha in 'UDemoCha.pas'
{SeriesForm},
  Basic in 'basic.pas' {BasicForm},
  Bubble in 'bubble.pas'
{BubbleForm},
  Gantt in 'Gantt.pas' {GanttForm},
  Lastvalu in 'Lastvalu.pas'
{DigitalForm},
  Linked in 'Linked.pas'
{LinkedTablesForm},
```

```
AddXY(XValues.Last+tmpX,Rando
m(100),'',clTeeColor);
  end;
end;

<-------> procedure
TTeeMainForm.Button2Click(Sende
r: TObject);
begin
  ShowForm(TDemoForm);
end;

<-------> procedure
TTeeMainForm.Chart2MouseUp(Se
nder: TObject; Button:
TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  Label10Click(Self);
end;

<-------> procedure
TTeeMainForm.Chart2MouseDown
(Sender: TObject;
  Button: TMouseButton; Shift:
TShiftState; X, Y: Integer);
begin
  Chart2.BevelOuter:=bvLowered;
end;

<-------> procedure
TTeeMainForm.Label10Click(Sende
r: TObject);
begin
  ShowForm(TTeeAboutForm);
  Chart2.BevelOuter:=bvRaised;
end;

<-------> procedure
TTeeMainForm.Button3Click(Sende
r: TObject);
begin
  ShowForm(TFeaturesForm);
end;

<-------> procedure
TTeeMainForm.Button4Click(Sende
r: TObject);
begin
```

```
  Ushapes in 'Ushapes.pas' {ShapesForm},
  Ustack in 'Ustack.pas' {StackedForm},
  Uylegend in 'Uylegend.pas' {LegendXYForm};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm(TTeeMainForm, TeeMainForm);
  Application.Run;
end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
{**********************************************}
{ TeeChart Delphi Component Library          }
{ Areas Demo                                  }
{ Copyright (c) 1995-2001 by David Berneda    }
{ All rights reserved                          }
{**********************************************}
unit stackare;

interface

{ This form shows 3 TAreaSeries components on same Chart.
  Areas can be Stacked or Stacked 100%
}
uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, Teengine, Series, Buttons, Chart, ExtCtrls,
  TeeProcs
  ;

type
  TAreasForm = class(TForm)
    Panel1: TPanel;
```

```
  LogLab in 'loglab.pas' {LogLabelsForm},
  Mulaxis in 'mulaxis.pas' {CustomAxisForm},
  Pie in 'Pie.pas' {PieForm},
  Specs in 'Specs.pas' {ChartSpecs},
  Sqlbars in 'Sqlbars.pas' {SQLBarsForm},
  Stackare in 'stackare.pas' {AreasForm},
  Tablepie in 'Tablepie.pas' {TablePieForm},
  Uanizoom in 'Uanizoom.pas' {FormAnimatedZoom},
  Uarrows in 'uarrows.pas' {ArrowsForm},
  Uaxislab in 'Uaxislab.pas' {AxisLabelsForm},
  UBitmap in 'Ubitmap.pas' {BitmapForm},
  ucolor in 'ucolor.pas' {ColoredForm},
  Ucrossh in 'ucrossh.pas' {CrossHairForm},
  Udbhoriz in 'Udbhoriz.pas' {DBHorizBarForm},
  udemutil in 'Udemutil.pas',
  Udraw in 'Udraw.pas' {DrawForm},
  UFast in 'Ufast.pas' {FastLineForm},
  UHighLo in 'uhighlo.pas' {HighLowForm},
  Ukeyboa in 'ukeyboa.pas' {KeyboardForm},
  Ulegend in 'Ulegend.pas' {LegendForm},
  Umain in 'Umain.pas' {MiniForm},
  UMetafil in 'umetafil.pas' {MetafileForm},
  Uoverbar in 'Uoverbar.pas' {OverBarForm},
  Upages in 'Upages.pas' {PagesForm},
  Uprint in 'Uprint.pas' {PrintForm},
  Uscroll in 'Uscroll.pas' {ScrollForm},
  Uscrollb in 'Uscrollb.pas' {ScrollBarForm},
```

```
  With Area do
  begin
    XValues.DateTime:=False;
    Clear;
    Old:=500+Random(1000);
    for t:=1 to 30 do
    begin
      Old:=Old+Random(50)-25;
      Add( Old,'',clTeeColor);
    end;
    Cursor:=crTeeHand;
  end;
 end;

begin
 tmpCounter:=-1;
 tmpSeries:=nil;
 CreateRandom(AreaSeries1);
 CreateRandom(AreaSeries2);
 CreateRandom(AreaSeries3);
end;

<-------> procedure TAreasForm.RadioGroup1Click(Sender: TObject);
begin
 { Change how areas are displayed. (Stacked, Stacked 100%) }
 AreaSeries1.MultiArea:=TMultiArea(RadioGroup1.ItemIndex);
end;

<-------> procedure TAreasForm.CheckBox1Click(Sender: TObject);
begin
 Chart1.View3D:=CheckBox1.Checked;  { <-- turn on/off 3d }
end;

<-------> procedure TAreasForm.Chart1MouseMove(Sender: TObject; Shift: TShiftState;
 X, Y: Integer);

 <-------> procedure HitSeries(ASeries:TChartSeries);
```

```
    RadioGroup1: TRadioGroup;
    Chart1: TChart;
    BitBtn1: TBitBtn;
    AreaSeries1: TAreaSeries;
    AreaSeries2: TAreaSeries;
    AreaSeries3: TAreaSeries;
    CheckBox1: TCheckBox;
    Label1: TLabel;
    CheckBox2: TCheckBox;
    Timer1: TTimer;
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure RadioGroup1Click(Sender: TObject);
    <-------> procedure CheckBox1Click(Sender: TObject);
    <-------> procedure Chart1MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    <-------> procedure CheckBox2Click(Sender: TObject);
    <-------> procedure Timer1Timer(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    tmpDelta,tmpCounter,tmpIndex:Longint;
    tmpSeries:TAreaSeries;
  end;

implementation

{$R *.dfm}

{ Some random values.... }
<-------> procedure TAreasForm.FormCreate(Sender: TObject);

 <-------> procedure CreateRandom(Area:TAreaSeries);
  var t:Longint;
      Old:Double;
  begin
```

```
    With tmpSeries.GetVertAxis do
tmpDelta:=Round(Abs(Maximum-
Minimum)/50.0);
    if Random(2)=1 then tmpDelta:=-
tmpDelta;
  end;
  inc(tmpCounter);

tmp:=tmpSeries.YValue[tmpIndex];
  if (tmp+tmpDelta)>0 then
tmpSeries.YValue[tmpIndex]:=tmp+
tmpDelta;
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

3
{******************************
**************}
{ TeeChart Delphi Component
Library          }
{ TArrowSeries Example
}
{ Copyright (c) 1995-2001 by David
Berneda    }
{ All rights reserved                    }
{******************************
**************}
unit uarrows;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, Teengine, Series,
ArrowCha, Chart, StdCtrls,
ExtCtrls,
  Buttons, TeeProcs;

type
  TArrowsForm = class(TForm)
    Panel1: TPanel;
    CheckBox1: TCheckBox;
    Chart1: TChart;
    ArrowSeries1: TArrowSeries;
    Timer1: TTimer;
```

```
  Var tmp:Longint;
  begin
    tmp:=ASeries.Clicked(x,y);
    if tmp<>-1 then
    begin
      Label1.Caption:=ASeries.Name;

Label1.Font.Color:=ASeries.SeriesC
olor;
    end;
  end;

var t:Longint;
begin
  Label1.Caption:='';
  for t:=0 to Chart1.SeriesCount-1 do
HitSeries(Chart1.Series[t]);

Label1.Visible:=Label1.Caption<>'';
end;

<-------> procedure
TAreasForm.CheckBox2Click(Send
er: TObject);
begin

Timer1.Enabled:=CheckBox2.Check
ed;
  if Timer1.Enabled then
RadioGroup1.ItemIndex:=1;
end;

<-------> procedure
TAreasForm.Timer1Timer(Sender:
TObject);
var tmp:Double;
begin
  if (tmpCounter=-1) or
(tmpCounter=5) then
  begin
    Case Random(3) of
      0: tmpSeries:=AreaSeries1;
      1: tmpSeries:=AreaSeries2;
      2: tmpSeries:=AreaSeries3;
    end;

tmpIndex:=Random(tmpSeries.Cou
nt);
    tmpCounter:=0;
```

```
    x1:=Random( 300 ) - 150.0;
    if x1<50 then x1:=50;
    x1:=x1+x0;

    y1:=Random( 300 ) - 150.0;
    if y1<50 then y1:=50;
    y1:=y1+y0;

    AddArrow( x0,y0,x1,y1, '',
clTeeColor );
  end;
 end;
end;

<-------> procedure
TArrowsForm.CheckBox1Click(Sen
der: TObject);
begin

Timer1.Enabled:=CheckBox1.Check
ed;
end;

<-------> procedure
TArrowsForm.Timer1Timer(Sender
: TObject);
var t:Longint;
begin
 Timer1.Enabled:=False;
 With ArrowSeries1 do
 Begin
  for t:=0 to Count-1 do
  Begin

StartXValues[t]:=StartXValues[t]+R
andom(100)-50.0;

StartYValues[t]:=StartYValues[t]+R
andom(100)-50.0;
    EndXValues[t]
:=EndXValues[t]+Random(100)-
50.0;
    EndYValues[t]
:=EndYValues[t]+Random(100)-
50.0;
  End;
  Repaint;
 End;
 Timer1.Enabled:=True;
```

```
   BitBtn3: TBitBtn;
   Memo1: TMemo;
   <-------> procedure
FormCreate(Sender: TObject);
   <-------> procedure
CheckBox1Click(Sender: TObject);
   <-------> procedure
Timer1Timer(Sender: TObject);
  private
   { Private declarations }
  public
   { Public declarations }
   <-------> procedure
AddRandomArrows;
  end;

implementation

{$R *.dfm}

<-------> procedure
TArrowsForm.FormCreate(Sender:
TObject);
begin
 With ArrowSeries1 do
 Begin
   ArrowWidth:=32;
   ArrowHeight:=24;
   ColorEachPoint:=True;
   XValues.DateTime:=False;
   YValues.DateTime:=False;

   AddRandomArrows;
 end;
end;

<-------> procedure
TArrowsForm.AddRandomArrows;
var x0,y0,x1,y1:Double;
   t:Longint;
begin
 With ArrowSeries1 do
 Begin
   Clear;
   for t:=1 to 40 do
   begin
     x0:=Random( 1000 );
     y0:=Random( 1000 );

```

```
   any Font size adjustment.
   TeeChart Axis will draw all labels
you specify.
}
uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, Chart, Series,
ExtCtrls, StdCtrls, Teengine,
Buttons,
  TeeProcs;

type
  TAxisLabelsForm = class(TForm)
    Chart1: TChart;
    LineSeries1: TLineSeries;
    Panel1: TPanel;
    RadioGroup1: TRadioGroup;
    PointSeries1: TPointSeries;
    BitBtn3: TBitBtn;
    Memo1: TMemo;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
RadioGroup1Click(Sender:
TObject);
    <-------> procedure
Chart1GetNextAxisLabel(Sender:
TChartAxis;
      LabelIndex: Longint; var
LabelValue: Double; var Stop:
Boolean);
  private
    { Private declarations }
  public
    { Public declarations }
    DefaultLabels:Boolean;
  end;

implementation

{$R *.dfm}

<-------> procedure
TAxisLabelsForm.FormCreate(Send
er: TObject);
var t:Longint;
begin
```

```
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
4
{*******************************
************}
{    TeeChart. TChart Component
}
{ Copyright (c) 1995-2001 by David
Berneda }
{    All Rights Reserved              }
{*******************************
************}
unit uaxislab;

interface

{ This Sample Project shows how to
set the Axis Labels at specific Axis
  positions.
  The key is the
Chart.OnGetNextAxisLabel event.
  This event is called continuosly for
each Axis Label until user decides
  to stop.
  At each call, you can specify the
exact Axis value where a Label must
be
  drawn.

  In this example, this event is used
to set the BottomAxis labels to the
  first (1) day in month:   1/1/96,
2/1/96, 3/1/96..... 12/1/96

  This don't needs necessarily to be
datetime values.
  You can also set the Axis Labels in
non-datetime axis.

  WARNING:
    Remember to set the Stop boolean
variable to TRUE when no more
labels are
    needed.
    Remember also that using this
event will NOT calculate Label used
space or
```

Chart1.BottomAxis.Increment := DateTimeStep[dtOneMonth];
and...
Chart1.BottomAxis.ExactDateTime := True ;
Eliminates the need for the following code.
}
{ ***************************** }
{ LabelValue has the "candidate" value where the Axis label will be painted. }
DecodeDate(LabelValue,year,month, day);
{ we force that value to be the first day in month }
Day:=1;
Month:=Month+1;
if Month>12 then
Begin
Month:=1;
Year:=Year+1;
end;
{ Then we set the preferred Label value }
LabelValue:=EncodeDate(year,mont h,day);
end
else
if Sender=Chart1.LeftAxis then
Begin
{ In this example, we want the Vertical Left Axis to show
labels only for positive values, starting at zero and
with 250 label increment.
}
if LabelValue>=250 then LabelValue:=LabelValue+250
else LabelValue:=250;

DefaultLabels:=False; { <-- boolean variable to show or not the demo }
LineSeries1.Clear;
PointSeries1.Clear;
for t:=1 to 100 do
Begin
LineSeries1.AddXY(Date+t, 200+Random(700),'',clTeeColor); { <-- some random points }
PointSeries1.AddXY(Date+t, 200+Random(700),'',clTeeColor);
end;
end;
<-------> procedure TAxisLabelsForm.RadioGroup1Clic k(Sender: TObject);
begin
{ Choose between default and custom labeling. }
DefaultLabels:=RadioGroup1.ItemI ndex=0;
Chart1.Repaint; { <-- repaint chart to see changes }
end;
<-------> procedure TAxisLabelsForm.Chart1GetNextA xisLabel(Sender: TChartAxis;
LabelIndex: Longint; var LabelValue: Double; var Stop: Boolean);
var year,month,day:Word;
begin
if not DefaultLabels then
Begin
if Sender=Chart1.BottomAxis then
Begin
{ ***************************** }
{ WARNING:
Setting this axis increment:

```
    Writing: TPointSeries;
    Reading: TPointSeries;
    BitBtn1: TBitBtn;
    RadioGroup1: TRadioGroup;
    BitBtn3: TBitBtn;
    CheckBox2: TCheckBox;
    SaveDialog1: TSaveDialog;
    BitBtn2: TBitBtn;
    Winter: TLineSeries;
    Summer: TLineSeries;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
CheckBox1Click(Sender: TObject);
    <-------> procedure
Timer1Timer(Sender: TObject);
    <-------> procedure
FormResize(Sender: TObject);
    <-------> procedure
FormShow(Sender: TObject);
    <-------> procedure
BitBtn1Click(Sender: TObject);
    <-------> procedure
CheckBox2Click(Sender: TObject);
    <-------> procedure
BitBtn2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}

Uses Printers;

<-------> procedure
TBasicForm.FormCreate(Sender:
TObject);
var t:Longint;
begin
  Summer.FillSampleValues(20); { <--
- Some random points }
  Winter.FillSampleValues(20); { <--
Some random points }

  { Will need a Bar Series with
```

```
    End;
    { we want more labels !! }
    Stop:=False;
  end;
end;


end.
\\\\\\\\\\\\\\\\\\\\\\\\\
5
{*******************************
***************}
{ TeeChart Delphi Component
Library         }
{ Basic Series Types Demo
}
{ Copyright (c) 1995-2001 by David
Berneda    }
{ All rights reserved                    }
{*******************************
***************}
unit basic;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, Chart, Series,
ExtCtrls, StdCtrls, Bubble,
  BubbleCh, Teengine, Buttons,
teeprocs;

type
  TBasicForm = class(TForm)
    Chart1: TChart;
    Panel1: TPanel;
    CheckBox1: TCheckBox;
    Timer1: TTimer;
    BarSeries1: TBarSeries;
    South: TAreaSeries;
    Speaking: TPointSeries;
    Panel2: TPanel;
    Chart2: TChart;
    Label1: TLabel;
    Chart3: TChart;
    Chart4: TChart;
    North: TAreaSeries;
```

```
          '',clTeeColor);
    end;
  end;

Var tmpLabel:String;
    tmpColor:TColor;
begin
  Animate(Summer);
  Animate(Winter);
  { BarSeries1 has special treatment to animate }
  With BarSeries1 do
  Begin
    tmpLabel:=XLabel[0];
    tmpColor:=ValueColor[0];
    Delete(0);
    AddXY(XValues.Last+1,Random(1000),tmpLabel,tmpColor);

    { Change Bar Style randomly... }
    if Random(10)<1 then
      BarStyle:=TBarStyle(Random(1+Ord(High(TBarStyle))));
  end;
  Animate(South);
  Animate(North);
  Animate(Speaking);
  Animate(Reading);
  Animate(Writing);

  { Change Pointer Style randomly... }
  if Random(10)<1 then
  With Speaking do
    Pointer.Style:=TSeriesPointerStyle(Random(1+Ord(High(TSeriesPointerStyle))));
end;

{ Realign the four charts }
<-------> procedure TBasicForm.FormResize(Sender: TObject);
Var w,h:Longint;
begin
  { Top and bottom Panel positioning
```

```
special random values,
    so we dont call the standard FillSampleValues method }
  With BarSeries1 do
  for t:=1 to 12 do
Add(Random(1000),ShortMonthNames[t],GetDefaultColor(t));

  South.FillSampleValues(20); { <-- Some random points }
  North.FillSampleValues(20); { <-- Some random points }

  Speaking.FillSampleValues(20); { <-- Some random points }
  Reading.FillSampleValues(20); { <-- Some random points }
  Writing.FillSampleValues(20); { <-- Some random points }

{  Force to resize the four charts }
  FormResize(Self);
end;

<-------> procedure TBasicForm.CheckBox1Click(Sender: TObject);
begin
  Timer1.Enabled:=CheckBox1.Checked;
end;

<-------> procedure TBasicForm.Timer1Timer(Sender: TObject);

  <-------> procedure Animate(Series:TChartSeries);
  Begin
    With Series do
    Begin
      Delete(0); { <-- remove the first point }
      { Add a new random point }
      AddXY( XValues.Last+1,
             YValues.Last+(Random(ChartSamplesMax)-(ChartSamplesMax/2)),
```

```
{ <-- Force Horizontal paper }
  try
    Printer.BeginDoc;       { <-- start printer job }
    try
      Printer.Title:='TeeChart Printing Demo';

      Case RadioGroup1.ItemIndex of
        0: Begin { screen proportional }
             Chart1.PrintResolution:= 0;
             Chart2.PrintResolution:= 0;
             Chart3.PrintResolution:= 0;
             Chart4.PrintResolution:= 0;
           End;
        1: Begin { thin lines and small fonts }
             Chart1.PrintResolution:= -100;
             Chart2.PrintResolution:= -100;
             Chart3.PrintResolution:= -100;
             Chart4.PrintResolution:= -100;
           End;
      end;

      { Print the four charts, each one at a different paper position }

      { CALCULATE HORIZONTAL MARGIN }
      tmpW:=Printer.PageWidth;
      tmpWMargin:=Round(5.0*tmpW/100.0); { <-- 5% margins }
      tmpW:=tmpW-2*tmpWMargin; { <-- left and right margins }
      tmpW:=tmpW div 2; { half height for left and right charts }

      { CALCULATE VERTICAL MARGIN }
      tmpH:=Printer.PageHeight;
      tmpHMargin:=Round(5.0*tmpH/100.0);  { <-- 5% margins }
      tmpH:=tmpH-2*tmpHMargin;  {
```

```
}
  w:=ClientWidth-4-Panel1.Width;
  Panel2.Width:=w-4;
  Panel1.Height:=ClientHeight-Panel2.Height-4;
  h:=Panel1.Height;
  { Top Left Chart }
  Chart1.SetBounds(Panel1.Width,Panel2.Height,w div 2,h div 2);
  { Bottom Left Chart }
  Chart2.SetBounds(Panel1.Width,Panel2.Height+Chart1.Height,w div 2,h div 2);
  { Top Right Chart }
  Chart3.SetBounds(Panel1.Width+Chart1.Width+1,Chart1.Top,w div 2,h div 2);
  { Bottom Right Chart }
  Chart4.SetBounds(Chart3.Left,Chart2.Top,w div 2,h div 2);
end;

<-------> procedure TBasicForm.FormShow(Sender: TObject);
begin
  FormResize(Self); { <-- to align charts }
end;

<-------> procedure TBasicForm.BitBtn1Click(Sender: TObject);
Var tmpH,tmpW,tmpWMargin,tmpHMargin:Longint; { margins }
    OldOrientation:TPrinterOrientation;
begin
  Screen.Cursor := crHourGlass;
  OldOrientation:=Printer.Orientation;  { <-- save paper orientation }
  Printer.Orientation:=poLandscape;
```

```
restore cursor }
  end;
end;

<-------> procedure
TBasicForm.CheckBox2Click(Sende
r: TObject);
begin

Chart1.View3D:=CheckBox2.Check
ed;

Chart2.View3D:=CheckBox2.Check
ed;

Chart3.View3D:=CheckBox2.Check
ed;

Chart4.View3D:=CheckBox2.Check
ed;
end;

<-------> procedure
TBasicForm.BitBtn2Click(Sender:
TObject);
var tmpH,tmpW:Longint;
begin
 { This code creates and stores a new
BITMAP file
   which contains the FOUR charts.
   Asks previously the user the BMP
filename.
 }
  with SaveDialog1 do
  begin
    if Execute then
    With TBitmap.Create do
    try
      { calculate bitmap size (2x2) }
      tmpW:=Chart1.Width;
      tmpH:=Chart1.Height;
      Width := 2*tmpW;
      Height:= 2*tmpH;
      { draw chart 1 }
      Chart1.BufferedDisplay:=False;

Chart1.Draw(Canvas,Rect(0,0,tmp
W,tmpH));
      Chart1.BufferedDisplay:=True;
```

```
<-- bottom and top margins }
      tmpH:=tmpH div 2; { half height
for top and bottom charts }

      { left / top chart }
      Chart1.PrintPartial( Rect(
tmpWMargin,tmpHMargin,

tmpWMargin+tmpW,tmpHMargin+
tmpH ) );

      { right / top chart }
      Chart3.PrintPartial( Rect(
tmpWMargin+tmpW,tmpHMargin,

tmpWMargin+2*tmpW,tmpHMargi
n+tmpH ) );

      { left / bottom chart }
      Chart2.PrintPartial( Rect(
tmpWMargin,tmpHMargin+tmpH,

tmpWMargin+tmpW,tmpHMargin+
2*tmpH ) );

      { right / bottom chart }
      Chart4.PrintPartial( Rect(
tmpWMargin+tmpW,tmpHMargin+
tmpH,

tmpWMargin+2*tmpW,tmpHMargi
n+2*tmpH ) );

      Printer.EndDoc; { <-- end job
and print !! }
    except
      on Exception do  { just in case an
error happens... }
      Begin
        Printer.Abort;
        Printer.EndDoc;
        Raise;
      end;
    end;
  finally

Printer.Orientation:=OldOrientatio
n;  { <-- restore paper orientation }
    Screen.Cursor:=crDefault; { <--
```

```
{ Bitmap images (TBitmap component) can be painted in several styles:

  pbmStretch : Bitmap is stretched to fit Chart Panel Rectangle
  pbmTile    : Bitmap is repeteadly painted without stretching.
  pbmCenter  : Bitmap is painted centered without stretching.

  You use the PanelBitmapMode property:

  Chart1.PanelBitmapMode := pbmTile ;
}

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, ExtCtrls, Chart, Series, StdCtrls, Teengine, Buttons,
  TeeProcs;

type
  TBitmapForm = class(TForm)
    Chart1: TChart;
    OpenDialog1: TOpenDialog;
    Panel1: TPanel;
    RadioGroup1: TRadioGroup;
    BitBtn1: TBitBtn;
    BitBtn3: TBitBtn;
    CheckBox1: TCheckBox;
    Series1: TBarSeries;
    Button1: TButton;
    Memo1: TMemo;
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure RadioGroup1Click(Sender: TObject);
    <-------> procedure BitBtn1Click(Sender: TObject);
    <-------> procedure CheckBox1Click(Sender: TObject);
    <-------> procedure Button1Click(Sender: TObject);
```

```

      { draw chart 2 }
      Chart2.BufferedDisplay:=False;

Chart2.Draw(Canvas,Rect(0,tmpH+1,tmpW,2*tmpH));
      Chart2.BufferedDisplay:=True;

      { draw chart 3 }
      Chart3.BufferedDisplay:=False;

Chart3.Draw(Canvas,Rect(tmpW+1,0,2*tmpW,tmpH));
      Chart3.BufferedDisplay:=True;

      { draw chart 4 }
      Chart4.BufferedDisplay:=False;

Chart4.Draw(Canvas,Rect(tmpW+1,tmpH+1,2*tmpW,2*tmpH));
      Chart4.BufferedDisplay:=True;
      SaveToFile(FileName);
    finally
      Free;
    end;
  end;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\
6
{*******************************************}
{    TeeChart. TChart Component           }
{ Copyright (c) 1995-2001 by David Berneda }
{    All Rights Reserved                   }
{*******************************************}
unit ubitmap;

interface

{ This example shows how to draw a custom Bitmap on a Panel Chart component }

```

```
x1.Checked;
end;

<-------> procedure TBitmapForm.Button1Click(Sender : TObject);
begin
  Chart1.BackImage:=nil;
end;


end.

\\\\\\\\\\\\\\\\\\\\\\

7
{*********************************************}
{ TeeChart Delphi Component Library          }
{ Bubble Series Type Demo
}
{ Copyright (c) 1995-2001 by David Berneda    }
{ All rights reserved                         }
{*********************************************}
unit bubble;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, Chart, Series, ExtCtrls, StdCtrls, BubbleCh,
  Teengine, Buttons, TeeProcs;

type
  TBubbleForm = class(TForm)
    Chart1: TChart;
    Panel1: TPanel;
    CheckBox1: TCheckBox;
    Timer1: TTimer;
    CheckBox2: TCheckBox;
    ComboBox1: TComboBox;
    Label1: TLabel;
    BitBtn3: TBitBtn;
```

```
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}

<-------> procedure TBitmapForm.FormCreate(Sender: TObject);
begin
 { This is to show something random in this example }
  Series1.FillSampleValues(10);
end;

<-------> procedure TBitmapForm.RadioGroup1Click(Sender: TObject);
begin

Chart1.BackImageMode:=TTeeBackImageMode(RadioGroup1.ItemIndex);
end;

<-------> procedure TBitmapForm.BitBtn1Click(Sender: TObject);
begin
  if OpenDialog1.Execute then
  begin
    RadioGroup1.Enabled:=False;

Chart1.BackImage.LoadFromFile(OpenDialog1.FileName);
    RadioGroup1.Enabled:=True;
  end;
end;

<-------> procedure TBitmapForm.CheckBox1Click(Sender: TObject);
begin

Chart1.BackImageInside:=CheckBo
```

for t:=1 to 100 do
BubbleSeries1.AddBubble(Date+t,
Random(ChartSamplesMax), { <-- y value }
ChartSamplesMax/(20+Random(25)), { <-- radius value }
'', { <-- label string }
GetDefaultColor(t)); { <-- color }
end;
<-------> procedure TBubbleForm.CheckBox1Click(Sender: TObject);
begin
Timer1.Enabled:=CheckBox1.Checked; { <-- on / off animation }
end;
<-------> procedure TBubbleForm.Timer1Timer(Sender : TObject);
Var tmpColor:TColor;
Begin
Timer1.Enabled:=False; { <-- stop the timer (this is optional) }
With BubbleSeries1 do
Begin
tmpColor:=ValueColor[0];
Delete(0); { <-- remove the first point }
{ Add a new random bubble }
AddBubble(XValues.Last+1, { <-- x value }
Random(ChartSamplesMax), { <-- y value }
ChartSamplesMax/(20+Random(25)), { <-- radius value }
'', { <-- label string }
tmpColor); {

CheckBox3: TCheckBox;
BubbleSeries1: TBubbleSeries;
ZoomInButton: TSpeedButton;
ZoomOutButton: TSpeedButton;
Memo1: TMemo;
<-------> procedure FormCreate(Sender: TObject);
<-------> procedure CheckBox1Click(Sender: TObject);
<-------> procedure Timer1Timer(Sender: TObject);
<-------> procedure CheckBox2Click(Sender: TObject);
<-------> procedure ComboBox1Change(Sender: TObject);
<-------> procedure CheckBox3Click(Sender: TObject);
function BubbleSeries1GetPointerStyle(Sender: TChartSeries;
ValueIndex: Longint): TSeriesPointerStyle;
<-------> procedure ZoomInButtonClick(Sender: TObject);
<-------> procedure ZoomOutButtonClick(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;
implementation
{$R *.dfm}
<-------> procedure TBubbleForm.FormCreate(Sender: TObject);
var t:Longint;
begin
ComboBox1.ItemIndex:=Ord(psCircle); { <-- Circled Bubbles by default }
BubbleSeries1.Clear;

```
kBox2.Checked; { switch on/off
Marks }
end;

<-------> procedure
TBubbleForm.ComboBox1Change(S
ender: TObject);
begin { the demo combobox1 allows
changing Bubble style }

BubbleSeries1.Pointer.Style:=TSerie
sPointerStyle(ComboBox1.ItemInde
x);
end;

<-------> procedure
TBubbleForm.CheckBox3Click(Sen
der: TObject);
begin
  Chart1.Repaint;
end;

function
TBubbleForm.BubbleSeries1GetPoi
nterStyle(Sender: TChartSeries;
  ValueIndex: Longint):
TSeriesPointerStyle;
begin
  if CheckBox3.Checked then

result:=TSeriesPointerStyle(Rando
m(Ord(High(TSeriesPointerStyle))))
  else

result:=BubbleSeries1.Pointer.Style;
end;

<-------> procedure
TBubbleForm.ZoomInButtonClick(
Sender: TObject);
begin
  Chart1.ZoomPercent(110);
end;

<-------> procedure
TBubbleForm.ZoomOutButtonClick
(Sender: TObject);
begin
  Chart1.ZoomPercent(90);
```

```
<-- color }
  end;
  if Random(100)<8 then
  begin
    if
ComboBox1.ItemIndex<ComboBox1
.Items.Count-1 then

ComboBox1.ItemIndex:=ComboBox
1.ItemIndex+1
    else
       ComboBox1.ItemIndex:=0;
    ComboBox1Change(Self);
  end;
  if (GetTickCount mod 1000)<=55
then
  begin
    with
BubbleSeries1.GetHorizAxis.Title do
     if Angle>=90 then Angle:=Angle-
90 else Angle:=270;

    with
BubbleSeries1.GetVertAxis.Title do
     if Angle>=90 then Angle:=Angle-
90 else Angle:=270;

    with BubbleSeries1.GetVertAxis
do
     if LabelsAngle>=90 then
LabelsAngle:=LabelsAngle-90 else
LabelsAngle:=270;

    with BubbleSeries1.GetHorizAxis
do
     if LabelsAngle>=90 then
LabelsAngle:=LabelsAngle-90 else
LabelsAngle:=270;
  end;
  Timer1.Enabled:=True;  { <--
restart the timer }
end;

<-------> procedure
TBubbleForm.CheckBox2Click(Sen
der: TObject);
begin

BubbleSeries1.Marks.Visible:=Chec
```

```
{$R *.dfm}
Uses TeeAbout;

end.
\\\\\\\\\\\\\\\\\\\\\
9
{*******************************
************}
{ TeeChart                        }
{ TColoredForm Example
}
{ Copyright (c) 1995-2001 by David
Berneda }
{    All Rights Reserved                }
{*******************************
************}
unit ucolor;

interface

uses
  Wintypes,WinProcs, Messages,
SysUtils, Classes, Graphics,
Controls, Forms, Dialogs,
  Teengine, Series, ExtCtrls, Chart,
StdCtrls, Buttons, teeprocs;

type
  TColoredForm = class(TForm)
    Chart1: TChart;
    Panel1: TPanel;
    CheckBox1: TCheckBox;
    BitBtn2: TBitBtn;
    LineSeries1: TLineSeries;
    PointSeries1: TPointSeries;
    Memo1: TMemo;
    <--------> procedure
FormCreate(Sender: TObject);
    <--------> procedure
CheckBox1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}
```

```
end;



end.

\\\\\\\\\\

8

{*******************************
***************}
{ TeeChart Delphi Component
Library          }
{ Chart Specs Form Demo
}
{ Copyright (c) 1995-2001 David
Berneda      }
{ All rights reserved                   }
{*******************************
***************}
unit specs;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, StdCtrls, Buttons,
ExtCtrls;

type
  TChartSpecs = class(TForm)
    Memo1: TMemo;
    Panel2: TPanel;
    Label1: TLabel;
    Image2: TImage;
    BitBtn1: TBitBtn;
    Panel1: TPanel;
    Memo2: TMemo;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

```

```
Berneda    }
{ All rights reserved                    }
{******************************
***************}
unit ucrossh;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, StdCtrls, Teengine,
Series, ExtCtrls, Chart, Buttons,
  TeeProcs;

type
  TCrossHairForm = class(TForm)
    Chart1: TChart;
    LineSeries1: TLineSeries;
    Panel1: TPanel;
    Label1: TLabel;
    BitBtn1: TBitBtn;
    Label2: TLabel;
    CheckBox1: TCheckBox;
    BitBtn2: TBitBtn;
    Memo1: TMemo;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
Chart1MouseMove(Sender:
TObject; Shift: TShiftState; X,
      Y: Integer);
    <-------> procedure
LineSeries1AfterDrawValues(Sende
r: TObject);
    <-------> procedure
CheckBox1Click(Sender: TObject);
    <-------> procedure
BitBtn2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    OldX,OldY:Longint;
    CrossHairColor:TColor;
    CrossHairStyle:TPenStyle;
  end;


<-------> procedure
TColoredForm.FormCreate(Sender:
TObject);

  <-------> procedure
AddColors(Series:TChartSeries);
  var step:Double;
      t:Longint;
  begin
    With Series,GetVertAxis do
    begin
      step:=(Maximum-
Minimum)/10.0;
      for t:=0 to Count-1 do

ValueColor[t]:=GetDefaultColor(
Trunc((YValue[t]-Minimum)/step) );
    end;
  end;

begin
  LineSeries1.FillSampleValues(100);

PointSeries1.FillSampleValues(100);
  Chart1.LeftAxis.AdjustMaxMin;
  AddColors(LineSeries1);
  AddColors(PointSeries1);
end;

<-------> procedure
TColoredForm.CheckBox1Click(Sen
der: TObject);
begin

Chart1.LeftAxis.Inverted:=CheckBo
x1.Checked;
end;


end.
\\\\\\\\\\\\\\\\
10
{******************************
***************}
{ TeeChart Delphi Component
Library           }
{ Cross-Hair demo                        }
{ Copyright (c) 1995-2001 by David
```

```
  if (OldX<>-1) then
  begin
    DrawCross(OldX,OldY);  { draw
old crosshair }
    OldX:=-1;
  end;

  { check if mouse is inside Chart
rectangle }
  if PtInRect( Chart1.ChartRect,
Point(X-
Chart1.Width3D,Y+Chart1.Height3
D)  ) then
  begin
    DrawCross(x,y);  { draw crosshair
at current position }
    { store old position }
    OldX:=x;
    OldY:=y;
    { set label text }
    With LineSeries1 do
    begin
      GetCursorValues(tmpX,tmpY);
{ <-- get values under mouse cursor }

Label1.Caption:=GetVertAxis.Label
Value(tmpY)+
                     ' '+

GetHorizAxis.LabelValue(tmpX);
    end;
  end;
end;

<-------> procedure
TCrossHairForm.LineSeries1AfterD
rawValues(Sender: TObject);
begin
  OldX:=-1;  { Reset old mouse
position }
end;

<-------> procedure
TCrossHairForm.CheckBox1Click(S
ender: TObject);
begin
  if CheckBox1.Checked then
Chart1.Cursor:=crCross
                    else
```

```
implementation

{$R *.dfm}
Uses udemutil;

<-------> procedure
TCrossHairForm.FormCreate(Send
er: TObject);
begin
  LineSeries1.FillSampleValues(30);
{ <-- some random values }
  OldX:=-1;                { initialize
variables }
  CrossHairColor:=clYellow;
  CrossHairStyle:=psSolid;
end;

<-------> procedure
TCrossHairForm.Chart1MouseMov
e(Sender: TObject; Shift:
TShiftState; X,
  Y: Integer);

  { This <-------> procedure draws
the crosshair lines }
  <-------> procedure
DrawCross(AX,AY:Integer);
  begin
    With Chart1,Canvas do
    begin
      Pen.Color:=CrossHairColor;
      Pen.Style:=CrossHairStyle;
      Pen.Mode:=pmXor;
      Pen.Width:=1;
      MoveTo(ax,ChartRect.Top-
Height3D);
      LineTo(ax,ChartRect.Bottom-
Height3D);

MoveTo(ChartRect.Left+Width3D,a
y);

LineTo(ChartRect.Right+Width3D,
ay);
    end;
  end;

Var tmpX,tmpY:Double;
begin
```

```
   BitBtn1: TBitBtn;
   CheckBox2: TCheckBox;
   PointSeries1: TPointSeries;
   FastLineSeries1: TFastLineSeries;
   DrawGrid: TCheckBox;
   Memo1: TMemo;
   <-------> procedure
FormCreate(Sender: TObject);
   <-------> procedure
LineSeries1AfterDrawValues(Sende
r: TObject);
   <-------> procedure
CheckBox1Click(Sender: TObject);
   <-------> procedure
Timer1Timer(Sender: TObject);
   <-------> procedure
CheckBox2Click(Sender: TObject);
   <-------> procedure
DrawGridClick(Sender: TObject);
  private
   { Private declarations }
  public
   { Public declarations }
   XPercent,YPercent:Integer;
  end;

implementation

{$R *.dfm}

<-------> procedure
TCustomAxisForm.FormCreate(Sen
der: TObject);

  <-------> procedure
CreateRandomPoints(Series:TChart
Series);
  var t,Old:Longint;
  begin
   With Series do
   begin
     Clear;
     Old:=Longint(Random(1000));
     for t:=1 to 100 do
     begin
       Inc(Old,Longint(Random(20))-
10);
       Add(Old,'',clTeeColor);
     end;
```

```
Chart1.Cursor:=crDefault;

Chart1.OriginalCursor:=Chart1.Cu
rsor;
end;

<-------> procedure
TCrossHairForm.BitBtn2Click(Send
er: TObject);
begin

CrossHairColor:=EditColor(Self,Cr
ossHairColor);
end;


end.
\\\\\\\\\\\\\\
11
{******************************
**************}
{ TeeChart Delphi Component
Library         }
{ Custom Axis Drawing Demo
}
{ Copyright (c) 1995-2001 by David
Berneda    }
{ All rights reserved                    }
{******************************
**************}
unit mulaxis;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, StdCtrls, ExtCtrls,
Teengine, Chart, Series, Buttons,
  TeeProcs;

type
  TCustomAxisForm = class(TForm)
    Chart1: TChart;
    LineSeries1: TLineSeries;
    Panel1: TPanel;
    CheckBox1: TCheckBox;
    Timer1: TTimer;
```

```
BottomAxis.CustomDraw(posaxis+10,posaxis+40,posaxis,DrawGrid.Checked);
  end;
end;

<-------> procedure TCustomAxisForm.CheckBox1Click(Sender: TObject);
begin
  Chart1.AxisVisible:=not CheckBox1.Checked;
  Timer1.Enabled:=CheckBox1.Checked;
end;

<-------> procedure TCustomAxisForm.Timer1Timer(Sender: TObject);
begin
  if XPercent<95 then Inc(XPercent,5)
              else XPercent:=5;
  if YPercent<97 then Inc(YPercent,3)
              else YPercent:=3;
  Chart1.Repaint;
end;

<-------> procedure TCustomAxisForm.CheckBox2Click(Sender: TObject);
begin
  Chart1.LeftAxis.Inverted:=CheckBox2.Checked;
  Chart1.BottomAxis.Inverted:=CheckBox2.Checked;
end;

<-------> procedure TCustomAxisForm.DrawGridClick(Sender: TObject);
begin
  Chart1.Repaint;
end;
```

```
   end;
  end;

begin
  XPercent:=50;
  YPercent:=50;
  Randomize;
  CreateRandomPoints(LineSeries1);
  CreateRandomPoints(PointSeries1);
  CreateRandomPoints(FastLineSeries1);
end;

<-------> procedure TCustomAxisForm.LineSeries1AfterDrawValues(Sender: TObject);
var posaxis:longint;
begin
  With Chart1 do
  begin
    { Calculate axis position and draw... }
    PosAxis:=ChartRect.Left+Trunc(ChartWidth*YPercent/100.0);
    LeftAxis.CustomDraw(posaxis-10,posaxis-40,posaxis,DrawGrid.Checked);

    PosAxis:=ChartRect.Left+Trunc(ChartWidth*(100.0-YPercent)/100.0);
    LeftAxis.CustomDraw(posaxis-10,posaxis-40,posaxis,DrawGrid.Checked);

    PosAxis:=ChartRect.Top+Trunc(ChartHeight*XPercent/100.0);
    BottomAxis.CustomDraw(posaxis+10,posaxis+40,posaxis,DrawGrid.Checked);

    PosAxis:=ChartRect.Top+Trunc(ChartHeight*(100.0-XPercent)/100.0);
```

```
TObject);
   <-------> procedure
CheckBox1Click(Sender: TObject);
   <-------> procedure
CheckBox2Click(Sender: TObject);
   <-------> procedure
Timer1Timer(Sender: TObject);
   <-------> procedure
BitBtn3Click(Sender: TObject);
  private
   { Private declarations }
  public
   { Public declarations }
   DeltaZ:Integer;
   <-------> procedure RecalcMarks;
  end;

implementation

{$R *.dfm}

<-------> procedure
TDBHorizBarForm.FormShow(Sen
der: TObject);
begin
  DeltaZ:=-5;  { <-- this variable is
used for animation only }

  { Here we set the Series order }
  With DBChart1.SeriesList do
  Begin
   Items[0]:=HorizBarSeries2;
   Items[1]:=HorizBarSeries1;
  End;
end;

<-------> procedure
TDBHorizBarForm.RecalcMarks;
Begin
  { Here the Series Marks and
associated Axis are
   adjusted depending on MultiBar
(Stacked or not )
  }
  if
(HorizBarSeries1.MultiBar=mbStac
ked) or

(HorizBarSeries1.MultiBar=mbStac
```

```

end.

\\\\\\\\\\\\\\\\\\\\\
12
{*******************************
******************}
{   TeeChart.   THorizBarSeries
Data-Aware Demo  }
{   Copyright (c) 1995-2001 by David
Berneda     }
{*******************************
******************}
unit udbhoriz;

interface

{ This form shows Horizontal Bar
Series attached to a Table }

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, DB, DBTables,
Chart, Series, ExtCtrls, DbChart,
StdCtrls,
  Teengine, Buttons, teeprocs;

type
  TDBHorizBarForm = class(TForm)
    Table1: TTable;
    DBChart1: TDBChart;
    Panel1: TPanel;
    CheckBox1: TCheckBox;
    RadioGroup1: TRadioGroup;
    Timer1: TTimer;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    HorizBarSeries1:
THorizBarSeries;
    HorizBarSeries2:
THorizBarSeries;
    CheckBox2: TCheckBox;
    Memo1: TMemo;
    <-------> procedure
FormShow(Sender: TObject);
    <-------> procedure
RadioGroup1Click(Sender:
```

ecked; { <-- 3 dimensions }
end;
<-------> procedure TDBHorizBarForm.CheckBox2Click(Sender: TObject);
begin
Timer1.Enabled:=CheckBox2.Checked; { <-- start / stop animation }
end;
<-------> procedure TDBHorizBarForm.Timer1Timer(Sender: TObject);
var tmp:Integer;
begin
Timer1.Enabled:=False;
{ here is the animation code }
tmp:=DBChart1.Chart3DPercent+DeltaZ;
if tmp<3 then
Begin
DeltaZ:=5;
tmp:=3;
end;
if tmp>90 then
begin
DeltaZ:=-5;
tmp:=90;
end;
DBChart1.Chart3DPercent:=tmp;
DBChart1.Legend.TopPos:=tmp;
if Random(100)=1 then
HorizBarSeries1.BarStyle:=TBarStyle(Random(Ord(High(TBarStyle))));
if Random(100)=1 then
HorizBarSeries2.BarStyle:=TBarStyle(Random(Ord(High(TBarStyle))));
if Random(100)=1 then
RadioGroup1.ItemIndex:=Random(RadioGroup1.Items.Count);
Timer1.Enabled:=True;
end;

ked100) then
Begin
DBChart1.Series[0].Marks.Visible:=False;
DBChart1.Series[1].Marks.Visible:=True;
HorizBarSeries1.HorizAxis:=aTopAxis;
HorizBarSeries2.HorizAxis:=aTopAxis;
end
else
Begin
DBChart1.Series[0].Marks.Visible:=True;
DBChart1.Series[1].Marks.Visible:=True;
HorizBarSeries1.HorizAxis:=aBottomAxis;
HorizBarSeries2.HorizAxis:=aTopAxis;
end
end;
<-------> procedure TDBHorizBarForm.RadioGroup1Click(Sender: TObject);
begin
{ Change the Series Stacked }
HorizBarSeries1.MultiBar:=TMultiBar(RadioGroup1.ItemIndex);
RecalcMarks;
end;
<-------> procedure TDBHorizBarForm.CheckBox1Click(Sender: TObject);
begin
DBChart1.View3D:=CheckBox1.Ch

```
Chart1: TChart;
Series1: TLineSeries;
Series2: TLineSeries;
Series3: TLineSeries;
Series4: TLineSeries;
Chart2: TChart;
Series5: TBarSeries;
Series6: TBarSeries;
Series7: TBarSeries;
Panel2: TPanel;
Chart3: TChart;
Series8: THorizBarSeries;
Series9: THorizBarSeries;
Series10: THorizBarSeries;
Chart4: TChart;
Series11: TAreaSeries;
Series12: TAreaSeries;
Chart5: TChart;
Series13: TPointSeries;
Series14: TPointSeries;
Series15: TPointSeries;
Series16: TPointSeries;
Series17: TPointSeries;
Series18: TPointSeries;
Chart6: TChart;
Series19: TPieSeries;
Chart7: TChart;
Series20: TFastLineSeries;
Series21: TFastLineSeries;
Series22: TFastLineSeries;
Series23: TFastLineSeries;
Series24: TFastLineSeries;
Chart8: TChart;
Series26: TArrowSeries;
Series27: TArrowSeries;
Chart9: TChart;
Series28: TBubbleSeries;
Chart10: TChart;
Series29: TGanttSeries;
Chart11: TChart;
Series30: TChartShape;
Series31: TChartShape;
Series32: TChartShape;
Button2: TButton;
Panel3: TPanel;
ListBox1: TListBox;
CheckBox1: TCheckBox;
CheckBox2: TCheckBox;
```

```
<-------> procedure
TDBHorizBarForm.BitBtn3Click(Sender: TObject);
var tmp:TChartSeries;
begin
  { This code "swaps" the two series }
  With DBChart1.SeriesList do
  Begin
    tmp:=Items[0];
    Items[0]:=Items[1];
    Items[1]:=tmp;
  End;
  RecalcMarks;
  DBChart1.Repaint;
end;



end.

\\\\\\\\\\\\\\\\\\
13

{*********************************************}
{ TeeChart Delphi Component Library v3       }
{ Copyright (c) 1995-2001 by David Berneda    }
{ All rights reserved                         }
{*********************************************}
unit teebasic;
{$P-}  { <-- Delphi 1.0 compatibility }

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, ExtCtrls, TeeProcs, TeEngine, Chart, StdCtrls, Series,
  TeeShape, GanttCh, BubbleCh, ArrowCha, Buttons;

type
  TDemoForm = class(TForm)
    Notebook1: TNotebook;
```

```
    Image1: TImage;
    Timer1: TTimer;
    ScrollBar1: TScrollBar;
    Label1: TLabel;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    SpeedButton3: TSpeedButton;
    SpeedButton4: TSpeedButton;
    Label2: TLabel;
    Label3: TLabel;
    CheckBox3: TCheckBox;
    ComboBox1: TComboBox;
    ComboBox2: TComboBox;
    ScrollBar2: TScrollBar;
    Button3: TButton;
    CheckBox4: TCheckBox;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    ScrollBar3: TScrollBar;
    Label8: TLabel;
    <-------> procedure
ListBox1Click(Sender: TObject);
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
CheckBox1Click(Sender: TObject);
    <-------> procedure
CheckBox2Click(Sender: TObject);
    <-------> procedure
ScrollBar1Change(Sender:
TObject);
    <-------> procedure
Timer1Timer(Sender: TObject);
    <-------> procedure
Button2Click(Sender: TObject);
    <-------> procedure
BitBtn1Click(Sender: TObject);
    <-------> procedure
BitBtn2Click(Sender: TObject);
    <-------> procedure
SpeedButton1Click(Sender:
TObject);
    <-------> procedure
SpeedButton4Click(Sender:
TObject);
    <-------> procedure
SpeedButton2Click(Sender:
TObject);
    <-------> procedure
SpeedButton3Click(Sender:
TObject);
    <-------> procedure
CheckBox3Click(Sender: TObject);
    <-------> procedure
ComboBox1Change(Sender:
TObject);
    <-------> procedure
Image1Click(Sender: TObject);
    <-------> procedure
ScrollBar2Change(Sender:
TObject);
    <-------> procedure
Series28GetMarkText(Sender:
TChartSeries;
      ValueIndex: Integer; var
MarkText: string);
    <-------> procedure
Button3Click(Sender: TObject);
    <-------> procedure
CheckBox4Click(Sender: TObject);
    <-------> procedure
FormShow(Sender: TObject);
    <-------> procedure
ScrollBar3Change(Sender:
TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    tmpSeries,
    tmpIndex,
    tmpRandom,
    tmpTimes:Longint;
    Function TheChart:TChart;
    <-------> procedure
HorizScroll(Const Percent:Double);
    <-------> procedure
VertScroll(Const Percent:Double);
    <-------> procedure
ScrollAxis(Axis:TChartAxis; Const
Percent:Double);
  end;

implementation
```

ComboBox2.ItemIndex:=0;
ListBox1.ItemIndex :=0;
ListBox1Click(Self);
end;
<-------> procedure TDemoForm.CheckBox1Click(Sender: TObject);
begin
TheChart.View3D:=CheckBox1.Checked;
ScrollBar2.Enabled:=TheChart.View3D;
ScrollBar3.Enabled:=TheChart.View3D;
end;
<-------> procedure TDemoForm.CheckBox2Click(Sender: TObject);
begin
Timer1.Enabled:=CheckBox2.Checked;
ScrollBar1.Enabled:=Timer1.Enabled;
end;
<-------> procedure TDemoForm.ScrollBar1Change(Sender: TObject);
begin
Timer1.Interval:=ScrollBar1.Position;
end;
Function TDemoform.TheChart:TChart;
begin
with NoteBook1 do
result:=(Pages.Objects[PageIndex] as TPage).Controls[0] as TChart;
end;

{$R *.dfm}
Uses TeeAbout;
<-------> procedure TDemoForm.ListBox1Click(Sender: TObject);
var t:Longint;
begin
NoteBook1.PageIndex:=ListBox1.ItemIndex;
With TheChart do
begin
tmpTimes:=-1;
AnimatedZoom:=True;
AnimatedZoomSteps:=4;
for t:=0 to SeriesCount-1 do
With Series[t] do
FillSampleValues(NumSampleValues);
UndoZoom;
CheckBox1.Checked :=View3D;
ScrollBar2.Enabled :=View3D;
ScrollBar2.Position:=Chart3DPercent;
if Series[0] is TPieSeries then
ScrollBar3.Position:=View3DOptions.Elevation
else
ScrollBar3.Position:=View3DOptions.Rotation;
ScrollBar3.Enabled:=View3D;
end;
end;
<-------> procedure TDemoForm.FormCreate(Sender: TObject);
begin
tmpTimes:=-1;
ScrollBar1.Position:=Timer1.Interval;
ComboBox1.ItemIndex:=0;

```
                begin
RadiusValues[tmpIndex]:=tmpX;
                  if tmpRandom>0 then
tmpX:=XValues[tmpIndex]+1
                  else
tmpX:=XValues[tmpIndex]-1;
XValues[tmpIndex]:=tmpX;
                  if Random(10)<5 then
tmpX:=YValues[tmpIndex]+50
                  else
tmpX:=YValues[tmpIndex]-50;
YValues[tmpIndex]:=tmpX;
                end
                else tmpTimes:=0;
              end
              else
              With
Series[tmpSeries].MandatoryValueL
ist do
Value[tmpIndex]:=Value[tmpIndex]
+tmpRandom;
              Dec(tmpTimes);
              Repaint;
            end;
          end;
    5: (TheChart[0] as
TPieSeries).Rotate(358);
  end;
end;

<-------> procedure
TDemoForm.Button2Click(Sender:
TObject);
begin
  Close;
end;

<-------> procedure
TDemoForm.BitBtn1Click(Sender:
TObject);
begin
  TheChart.ZoomPercent(120);
  Button3.Enabled:=True;
end;

<-------> procedure
TDemoForm.Timer1Timer(Sender:
TObject);
var t:Longint;
   tmpX:Double;
begin
  With NoteBook1 do
  Case PageIndex of
    0,3,4,6:  With TheChart do
          begin
            for t:=0 to SeriesCount-1
do
            With Series[t] do
            begin
              tmpX:=XValues[1]-
XValues[0];
              Delete(0);
              AddXY(
XValues.Last+tmpX,
YValues.Last+Random(100)-
50,'',clTeeColor);
            end;
          end;
    1,2,8:    With TheChart do
          begin
            if (tmpTimes=-1) then
            begin
tmpSeries:=Random(SeriesCount);
              tmpIndex
:=Random(Series[tmpSeries].Count)
;
              tmpTimes
:=Random(10);
tmpRandom:=2*Round(Random(50
)-25.0);
            end;
            if tmpSeries<>-1 then
            begin
              if PageIndex=8 then
              With (Series[tmpSeries]
as TBubbleSeries) do
              begin
tmpX:=RadiusValues[tmpIndex]+tm
pRandom;
                if tmpX>=2 then
```

```
TDemoForm.SpeedButton1Click(Sender: TObject);
begin
  HorizScroll(10);
end;

<-------> procedure TDemoForm.SpeedButton4Click(Sender: TObject);
begin
  HorizScroll(-10);
end;

<-------> procedure TDemoForm.SpeedButton2Click(Sender: TObject);
begin
  VertScroll(-10);
end;

<-------> procedure TDemoForm.SpeedButton3Click(Sender: TObject);
begin
  VertScroll(10);
end;

<-------> procedure TDemoForm.CheckBox3Click(Sender: TObject);
begin
  (TheChart[0] as TPieSeries).Circled:=CheckBox3.Checked;
end;

<-------> procedure TDemoForm.ComboBox1Change(Sender: TObject);
begin
  With (TheChart[0] as TCustomBarSeries) do
   MultiBar:=TMultiBar((Sender as TComboBox).ItemIndex);
end;

<-------> procedure TDemoForm.Image1Click(Sender: TObject);
```

```
<-------> procedure TDemoForm.BitBtn2Click(Sender: TObject);
begin
  TheChart.ZoomPercent(80);
  Button3.Enabled:=True;
end;

<-------> procedure TDemoForm.ScrollAxis(Axis:TChartAxis; Const Percent:Double);
var Amount:Double;
begin
  With Axis do
  begin
    Amount:=-((Maximum-Minimum)/(100.0/Percent));
    SetMinMax(Minimum-Amount,Maximum-Amount);
  end;
end;

<-------> procedure TDemoForm.HorizScroll(Const Percent:Double);
begin
  ScrollAxis(TheChart.TopAxis,Percent);
  ScrollAxis(TheChart.BottomAxis,Percent);
  Button3.Enabled:=True;
end;

<-------> procedure TDemoForm.VertScroll(Const Percent:Double);
begin
  ScrollAxis(TheChart.LeftAxis,Percent);
  ScrollAxis(TheChart.RightAxis,Percent);
  Button3.Enabled:=True;
end;

<-------> procedure
```

```
Series11.Stairs:=CheckBox4.Checked;
Series12.Stairs:=CheckBox4.Checked;
end;

<-------> procedure TDemoForm.FormShow(Sender: TObject);
begin
  Timer1.Enabled:=True;  { <-- start animation }
end;

<-------> procedure TDemoForm.ScrollBar3Change(Sender: TObject);
begin
  if ScrollBar3.Enabled then
  With TheChart.View3DOptions do
  begin
    Orthogonal:=False;
    if TheChart[0] is TPieSeries then
       Elevation:=ScrollBar3.Position
    else
       Rotation:=ScrollBar3.Position;
  end;
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
14

{*********************************************}
{ TeeChart Delphi Component Library         }
{ Digital Series and Legend Last Values Demo  }
{ Copyright (c) 1995-2001 by David Berneda    }
{ All rights reserved                         }
{*********************************************}
{$P-}   { <-- VERY IMPORTANT WHEN USING OnGet.. EVENTS }
unit lastvalu;

begin
  Timer1.Enabled:=False;
  With TTeeAboutForm.Create(Self) do
  try
    ShowModal;
  finally
    Free;
  end;
  Timer1.Enabled:=True;
end;

<-------> procedure TDemoForm.ScrollBar2Change(Sender: TObject);
begin
TheChart.Chart3DPercent:=ScrollBar2.Position;
end;

<-------> procedure TDemoForm.Series28GetMarkText(Sender: TChartSeries;
  ValueIndex: Integer; var MarkText: string);
begin
   if ValueIndex=3 then MarkText:='USA' else
   if ValueIndex=5 then MarkText:='UK' else
   if ValueIndex=7 then MarkText:='Germany' else
                      MarkText:='';
end;

<-------> procedure TDemoForm.Button3Click(Sender: TObject);
begin
  TheChart.UndoZoom;
  Button3.Enabled:=False;
end;

<-------> procedure TDemoForm.CheckBox4Click(Sender: TObject);
begin
```

```
interface
{ This forms show 4 line series in "Stairs" mode.

 ( LineSeries1.Stairs := True ; )

  Legend Style is "lsLastValues" meaning the Legend will draw the
  Last value for each Series.
}

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, Chart, Series, ExtCtrls, StdCtrls, Teengine, Buttons,
  TeeProcs;

type
  TDigitalForm = class(TForm)
    Chart1: TChart;
    LineSeries1: TLineSeries;
    LineSeries2: TLineSeries;
    LineSeries3: TLineSeries;
    LineSeries4: TLineSeries;
    Panel1: TPanel;
    CheckBox1: TCheckBox;
    Timer1: TTimer;
    CheckBox2: TCheckBox;
    CheckBox3: TCheckBox;
    BitBtn3: TBitBtn;
    CheckBox4: TCheckBox;
    Memo1: TMemo;
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure CheckBox1Click(Sender: TObject);
    <-------> procedure Timer1Timer(Sender: TObject);
    <-------> procedure CheckBox2Click(Sender: TObject);
    <-------> procedure CheckBox3Click(Sender: TObject);
    <-------> procedure Chart1GetLegendText(Sender: TCustomAxisPanel;
      LegendStyle: TLegendStyle;
Index: Longint; var LegendText: String);
    <-------> procedure CheckBox4Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}

<-------> procedure TDigitalForm.FormCreate(Sender: TObject);
var t,tt:Longint;
begin
  Chart1.ApplyZOrder:=CheckBox4.Checked;    { ZOrder or not ZOrder... }
  Chart1.Legend.Inverted:=True;
  { Fill Series with random values }
  for t:=0 to Chart1.SeriesCount-1 do
  With Chart1.Series[t] do
  begin
    Clear;
    for tt:=1 to 100 do Add( 2*t+Random(2), '', clTeeColor);
  end;
end;

<-------> procedure TDigitalForm.CheckBox1Click(Sender: TObject);
begin
  Timer1.Enabled:=CheckBox1.Checked;  { start / stop animation }
end;

<-------> procedure TDigitalForm.Timer1Timer(Sender: TObject);
var t:Longint;
begin
  Timer1.Enabled:=False;  { <-- stop
```

```
t(Sender: TCustomAxisPanel;
  LegendStyle: TLegendStyle; Index:
Longint; var LegendText: String);
begin
  { we want to show the Series Title
as well as the Last Values }
  if LegendStyle=lsLastValues then
     LegendText:=LegendText+' -->
'+Chart1.Series[Index].Title;
end;

<-------> procedure
TDigitalForm.CheckBox4Click(Send
er: TObject);
begin

Chart1.ApplyZOrder:=CheckBox4.
Checked;
  Chart1.Repaint;
end;


end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\
15
{*******************************
************}
{    TeeChart. Draw Example
}
{ Copyright (c) 1995-2001 by David
Berneda }
{    All Rights Reserved                }
{*******************************
************}
unit udraw;

interface

{ THIS EXAMPLE SHOWS HOW
TO DRAW ADDITIONAL
CUSTOMIZED THINGS TO A
  CHART COMPONENT }
uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, Chart, Series,
ExtCtrls, Teengine, StdCtrls,
Buttons,
```

```
the timer }

  { Now, add a new point to each
Series }
  for t:=0 to Chart1.SeriesCount-1 do
  With Chart1.Series[t] do Add(
2*t+Random(2),'',clTeeColor);

  { Scroll the Horizontal Axis }
  With Chart1.BottomAxis do  { <--
with the Horizontal Axis... }
  Begin
    Automatic := False;       { <-- we
dont want automatic scaling }
    Maximum :=
LineSeries1.XValues.Last;
    Minimum := Maximum - 100;   {
we want to see the last 100 points
only }
  End;
  { re-start timer }
  Timer1.Enabled:=True;
end;

<-------> procedure
TDigitalForm.CheckBox2Click(Send
er: TObject);
begin

Chart1.View3D:=CheckBox2.Check
ed;
end;

<-------> procedure
TDigitalForm.CheckBox3Click(Send
er: TObject);
begin
  if CheckBox3.Checked then

Chart1.Legend.LegendStyle:=lsLast
Values
  else

Chart1.Legend.LegendStyle:=lsAuto
;
end;

<-------> procedure
TDigitalForm.Chart1GetLegendTex
```

```
<-------> procedure
TDrawForm.LineSeries1BeforeDrawValues(Sender: TObject);
Const
  MyColors:array[1..5] of TColor=
   ( clNavy,
     clGreen,
     clYellow,
     clRed,
     $00000080 { very red }
     );
var t,partial:Longint;
   tmpRect:TRect;
   YPosition:Longint;
   tmpYCenterValue:Double;
begin
  With Chart1 do
  Begin
   { we will divide the total chart
width by 5 }
   tmpRect:=ChartRect;
   tmpRect.Right:=tmpRect.Left;
   partial:=ChartWidth div 5;

   { change the brush style }

Canvas.Brush.Style:=bsDiagCross;
   Canvas.Pen.Style:=psClear;

   { for each section, fill with a
specific color }
   for t:=1 to 5 do
   Begin
     { adjust the rectangle dimension
}
     tmpRect.Right
:=tmpRect.Right+partial+1 ;

     { set the brush color }

Canvas.Brush.Color:=MyColors[t];

     { paint !!! }
     With tmpRect do
       Canvas.Rectangle(
Left+Width3D,Top-
Height3D,Right+Width3D,Bottom-
Height3D );
```

```
  TeeProcs;

type
  TDrawForm = class(TForm)
    Chart1: TChart;
    LineSeries1: TLineSeries;
    Panel1: TPanel;
    BitBtn3: TBitBtn;
    CheckBox1: TCheckBox;
    Timer1: TTimer;
    BitBtn1: TBitBtn;
    CheckBox2: TCheckBox;
    Memo1: TMemo;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
LineSeries1BeforeDrawValues(Sender: TObject);
    <-------> procedure
LineSeries1AfterDrawValues(Sender: TObject);
    <-------> procedure
Timer1Timer(Sender: TObject);
    <-------> procedure
CheckBox1Click(Sender: TObject);
    <-------> procedure
BitBtn1Click(Sender: TObject);
    <-------> procedure
CheckBox2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    Percent:Double;
    DeltaPercent:Integer;
  end;

implementation

{$R *.dfm}

<-------> procedure
TDrawForm.FormCreate(Sender:
TObject);
begin
  Percent:=50;  { <-- used for this
demo only }
  LineSeries1.FillSampleValues(20);
end;
```

ent*(MaxValue-MinValue)/100.0;
{ then calculate the Screen Pixel coordinate of the above value }
YPosition:=LeftAxis.CalcYPosValue (tmpYCenterValue);
{ change pen and draw the line }
Pen.Width:=3;
Pen.Style:=psSolid;
Pen.Color:=clRed;
MoveTo(ChartRect.Left,YPosition);
LineTo(ChartRect.Right,YPosition);
LineTo(ChartRect.Right+Width3D, YPosition-Height3D);
{ change font and draw some text above the line }
Font.Name:='Arial';
{ VERY IMPORTANT !!!!!! }
{ THIS IS NECESSARY IF YOU'RE GOING TO PRINT !!!! }
{ IT MAKES FONT SIZES TO WORK FINE BOTH AT SCREEN AND PRINTER. }
Font.Height:=-24; { <-- express font size in "Height", NOT "Size" }
Font.Color:=clYellow;
Font.Style:=[fsBold];
{ Set transparent background... }
Brush.Style:=bsClear;
{ Output some text... }
TextOut(ChartRect.Left+20,
YPosition-24 ,
'This is '+FloatToStr(tmpYCenterValue));
end;
end;

{ adjust rectangle }
tmpRect.Left:=tmpRect.Right;
end;
{ first calculate the middle vertical value (based on LineSeries points) }
With LineSeries1.YValues do
tmpYCenterValue:=MinValue+Percent*(MaxValue-MinValue)/100.0;
{ then calculate the Screen Pixel coordinate of the above value }
YPosition:=LeftAxis.CalcYPosValue (tmpYCenterValue);
With Canvas do
begin
{ change pen and draw the line }
Pen.Width:=3;
Pen.Style:=psSolid;
Pen.Color:=clRed;
MoveTo(ChartRect.Left,YPosition);
LineTo(ChartRect.Left+Width3D,YPosition-Height3D);
LineTo(ChartRect.Right+Width3D, YPosition-Height3D);
end;
end;
end;
<-------> procedure TDrawForm.LineSeries1AfterDrawValues(Sender: TObject);
Var YPosition:Longint;
tmpYCenterValue:Double;
begin
With Chart1,Canvas do
Begin
{ first calculate the middle vertical value (based on LineSeries points) }
With LineSeries1.YValues do
tmpYCenterValue:=MinValue+Perc

```
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
16

{******************************
************}
{    TeeChart. TChart Component
}
{ Copyright (c) 1995-2001 by David
Berneda }
{    All Rights Reserved              }
{******************************
************}
unit ufast;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, StdCtrls, ExtCtrls,
Teengine, Chart,Series, Buttons,
  TeeProcs;

type
  TFastLineForm = class(TForm)
    Chart1: TChart;
    Panel1: TPanel;
    Button1: TButton;
    CheckBox1: TCheckBox;
    CheckBox2: TCheckBox;
    FastLineSeries1: TFastLineSeries;
    FastLineSeries2: TFastLineSeries;
    CheckBox3: TCheckBox;
    BitBtn3: TBitBtn;
    Memo1: TMemo;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
Button1Click(Sender: TObject);
    <-------> procedure
CheckBox2Click(Sender: TObject);
    <-------> procedure
CheckBox1Click(Sender: TObject);
    <-------> procedure
CheckBox3Click(Sender: TObject);
  private
    { Private declarations }
  public
```

```
<-------> procedure
TDrawForm.Timer1Timer(Sender:
TObject);
begin
  if Percent+DeltaPercent>100 then
  begin
    Percent:=100;
    DeltaPercent:=-5;
  end
  else
  if Percent+DeltaPercent<0 then
  begin
    Percent:=0;
    DeltaPercent:=5;
  end
  else
Percent:=Percent+DeltaPercent;
  Chart1.Repaint;
end;

<-------> procedure
TDrawForm.CheckBox1Click(Sende
r: TObject);
begin

Timer1.Enabled:=CheckBox1.Check
ed;
  DeltaPercent:=5;
end;

<-------> procedure
TDrawForm.BitBtn1Click(Sender:
TObject);
begin
 { try with and without this --->
Chart1.PrintResolution := -100; }
  Chart1.PrintLandscape;
end;

<-------> procedure
TDrawForm.CheckBox2Click(Sende
r: TObject);
begin

Chart1.View3D:=CheckBox2.Check
ed;
end;

end.
```

```
  Chart1.AnimatedZoom:=False;
  t1:=GetTickCount;
  for t:=1 to 30 do
Chart1.ZoomPercent(105);  { 5%
zoom in }
  for t:=1 to 30 do
Chart1.ZoomPercent(95); { 5%
zoom out }
  t2:=GetTickCount;
  Chart1.AnimatedZoom:=True;
  Chart1.UndoZoom;
 finally
  Screen.Cursor:=crDefault;
 end;
 Showmessage('Time to plot 2000
points'+#13+
           '61 times: '+#13+
           IntToStr(t2-t1)+'
milliseconds.');
end;

<-------> procedure
TFastLineForm.CheckBox2Click(Se
nder: TObject);
begin

Chart1.AxisVisible:=CheckBox2.Ch
ecked;
end;

<-------> procedure
TFastLineForm.CheckBox1Click(Se
nder: TObject);
begin
 { Setting this to False speeds up
drawing
  with maximized charts on some
video drivers }

Chart1.BufferedDisplay:=CheckBox
1.Checked;
end;

<-------> procedure
TFastLineForm.CheckBox3Click(Se
nder: TObject);
begin

Chart1.ClipPoints:=CheckBox3.Che
  { Public declarations }
 end;

implementation

{$R *.dfm}

<-------> procedure
TFastLineForm.FormCreate(Sender
: TObject);
var t,tmpRandom:Longint;
begin
 { This can speed up things a little...
}
 TeeEraseBack:=False;

 { Make the chart flicker by default,
only for this demo }
 Chart1.BufferedDisplay:=False;

 Chart1.View3d:=False;
 Chart1.Legend.Visible:=False;
 Chart1.Title.Visible:=False;
 Chart1.Foot.Visible:=False;

 { some speed improvement if... }
 TeeDefaultCapacity:=2000;

 { 2000 random points }
 Randomize;
 for t:=1 to 1000 do
 begin
   tmpRandom:=Random(Abs(500-
t))-(Abs(500-t) div 2);
   FastLineSeries1.Add(1000-
t+tmpRandom,'',clTeeColor);

FastLineSeries2.Add(t+tmpRandom,
'',clTeeColor);
 end;
end;

<-------> procedure
TFastLineForm.Button1Click(Sende
r: TObject);
var t1,t2,t:Longint;
begin
 Screen.Cursor:=crHourGlass;
 try
```

```
    BitBtn21: TBitBtn;
    Panel2: TPanel;
    Label21: TLabel;
    Label22: TLabel;
    Label23: TLabel;
    Label24: TLabel;
    Label25: TLabel;
    Label26: TLabel;
    Label27: TLabel;
    Label28: TLabel;
    Label29: TLabel;
    Label30: TLabel;
    Label31: TLabel;
    Label32: TLabel;
    Label33: TLabel;
    Label34: TLabel;
    Label35: TLabel;
    Label36: TLabel;
    Label37: TLabel;
    Label38: TLabel;
    Label39: TLabel;
    Label40: TLabel;
    BitBtn22: TBitBtn;
    Label14: TLabel;
    Panel1: TPanel;
    <-------> procedure BitBtn3Click(Sender: TObject);
    <-------> procedure BitBtn5Click(Sender: TObject);
    <-------> procedure BitBtn6Click(Sender: TObject);
    <-------> procedure BitBtn7Click(Sender: TObject);
    <-------> procedure BitBtn2Click(Sender: TObject);
    <-------> procedure BitBtn8Click(Sender: TObject);
    <-------> procedure BitBtn9Click(Sender: TObject);
    <-------> procedure BitBtn10Click(Sender: TObject);
    <-------> procedure BitBtn11Click(Sender: TObject);
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure BitBtn12Click(Sender: TObject);
    <-------> procedure BitBtn13Click(Sender: TObject);
```

```
cked;
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
17
{*********************************************}
{ TeeChart Delphi Component Library         }
{ Main Form Demo                            }
{ Copyright (c) 1996-2001 by David Berneda   }
{ All rights reserved                       }
{*********************************************}
unit features;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, Buttons, StdCtrls, ExtCtrls;

type
  TFeaturesForm = class(TForm)
    BitBtn5: TBitBtn;
    BitBtn6: TBitBtn;
    BitBtn7: TBitBtn;
    BitBtn3: TBitBtn;
    BitBtn9: TBitBtn;
    BitBtn11: TBitBtn;
    BitBtn10: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn8: TBitBtn;
    BitBtn12: TBitBtn;
    BitBtn13: TBitBtn;
    BitBtn14: TBitBtn;
    BitBtn4: TBitBtn;
    BitBtn15: TBitBtn;
    BitBtn16: TBitBtn;
    BitBtn17: TBitBtn;
    BitBtn18: TBitBtn;
    BitBtn19: TBitBtn;
    BitBtn20: TBitBtn;
```

```
  end;
End;

<-------> procedure
TFeaturesForm.BitBtn3Click(Sende
r: TObject);
begin
  ShowForm(TMiniForm);
end;

<-------> procedure
TFeaturesForm.BitBtn5Click(Sende
r: TObject);
begin
  ShowForm(TLegendForm);
end;

<-------> procedure
TFeaturesForm.BitBtn6Click(Sende
r: TObject);
begin
  ShowForm(TBitmapForm);
end;

<-------> procedure
TFeaturesForm.BitBtn7Click(Sende
r: TObject);
begin
  ShowForm(TDrawForm);
end;

<-------> procedure
TFeaturesForm.BitBtn2Click(Sende
r: TObject);
begin
  ShowForm(TScrollForm);
end;

<-------> procedure
TFeaturesForm.BitBtn8Click(Sende
r: TObject);
begin
  ShowForm(TShapesForm);
end;

<-------> procedure
TFeaturesForm.BitBtn9Click(Sende
r: TObject);
begin
```

```
    <-------> procedure
BitBtn14Click(Sender: TObject);
    <-------> procedure
BitBtn4Click(Sender: TObject);
    <-------> procedure
BitBtn15Click(Sender: TObject);
    <-------> procedure
BitBtn16Click(Sender: TObject);
    <-------> procedure
BitBtn17Click(Sender: TObject);
    <-------> procedure
BitBtn18Click(Sender: TObject);
    <-------> procedure
BitBtn19Click(Sender: TObject);
    <-------> procedure
BitBtn20Click(Sender: TObject);
    <-------> procedure
BitBtn21Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    <-------> procedure
ShowForm(AFormClass:TFormClas
s);
  end;

implementation

{$R *.dfm}
Uses
UAxisLab,UBitmap,UDraw,ULegen
d,UMain,UOverBar,MulAxis,

UPages,UPrint,UScroll,UShapes,UY
Legend,LastValu,UColor,

UMetafil,UAniZoom,UScrollB,UCro
ssH,UKeyboa,LogLab;

<-------> procedure
TFeaturesForm.ShowForm(AForm
Class:TFormClass);
Begin
  With AFormClass.Create(Self) do
  try
    ShowModal;
  finally
    Free;
```

```
TFeaturesForm.BitBtn14Click(Send
er: TObject);
begin
  ShowForm(TDigitalForm);
end;

<-------> procedure
TFeaturesForm.BitBtn4Click(Sende
r: TObject);
begin

ShowForm(TFormAnimatedZoom);
end;

<-------> procedure
TFeaturesForm.BitBtn15Click(Send
er: TObject);
begin
  ShowForm(TMetafileForm);
end;

<-------> procedure
TFeaturesForm.BitBtn16Click(Send
er: TObject);
begin
  ShowForm(TCustomAxisForm);
end;

<-------> procedure
TFeaturesForm.BitBtn17Click(Send
er: TObject);
begin
  ShowForm(TColoredForm);
end;

<-------> procedure
TFeaturesForm.BitBtn18Click(Send
er: TObject);
begin
  ShowForm(TScrollBarForm);
end;

<-------> procedure
TFeaturesForm.BitBtn19Click(Send
er: TObject);
begin
  ShowForm(TCrossHairForm);
end;

  ShowForm(TOverBarForm);
end;

<-------> procedure
TFeaturesForm.BitBtn10Click(Send
er: TObject);
begin
  ShowForm(TPrintForm);
end;

<-------> procedure
TFeaturesForm.BitBtn11Click(Send
er: TObject);
begin
  ShowForm(TPagesForm);
end;

<-------> procedure
TFeaturesForm.FormCreate(Sender
: TObject);
begin
  if Screen.Width<800 then
    ShowMessage('Warning:
'+#13+#10+
            'This Demo is best viewed
with a Screen'+#13+#10+
            'resolution of 800x600 or
greater,'+#13+#10+
            'and a Color Depth of 256
or greater.'+#13+#10+
            '16K Colors is also better
than 256 Colors.');
end;

<-------> procedure
TFeaturesForm.BitBtn12Click(Send
er: TObject);
begin
  ShowForm(TAxisLabelsForm);
end;

<-------> procedure
TFeaturesForm.BitBtn13Click(Send
er: TObject);
begin
  ShowForm(TLegendXYForm);
end;

<-------> procedure
```

```
   LineSeries3: TLineSeries;
   LineSeries4: TLineSeries;
   LineSeries5: TLineSeries;
   PointSeries1: TPointSeries;
   Label1: TLabel;
   SpinEdit1: TSpinEdit;
   BitBtn2: TBitBtn;
   BitBtn1: TBitBtn;
   BitBtn3: TBitBtn;
   Memo1: TMemo;
   <-------> procedure
FormCreate(Sender: TObject);
   <-------> procedure
CheckBox1Click(Sender: TObject);
   <-------> procedure
CheckBox2Click(Sender: TObject);
   <-------> procedure
SpinEdit1Change(Sender: TObject);
   <-------> procedure
BitBtn1Click(Sender: TObject);
   <-------> procedure
BitBtn2Click(Sender: TObject);
  private
   { Private declarations }
  public
   { Public declarations }
  end;

implementation

{$R *.dfm}

<-------> procedure
TFormAnimatedZoom.FormCreate(
Sender: TObject);

  <-------> procedure
RandomValues(ASeries:TChartSeri
es);
  Var t,tmp:Longint;
  Begin
   With ASeries do
   Begin
     tmp:=0;
     Clear;
     for t:=1 to 50 do
     Begin
       tmp:=tmp+Random(10000)-
5000;
```

```
<-------> procedure
TFeaturesForm.BitBtn20Click(Send
er: TObject);
begin
  ShowForm(TKeyboardForm);
end;

<-------> procedure
TFeaturesForm.BitBtn21Click(Send
er: TObject);
begin
  ShowForm(TLogLabelsForm);
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\\\\\\\\\\
18

{*******************************
************}
{    TeeChart. TChart Component
}
{ Copyright (c) 1995-2001 by David
Berneda }
{    All Rights Reserved              }
{*******************************
************}
unit uanizoom;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, Teengine, Series,
ExtCtrls, Chart, StdCtrls, Spin,
  Buttons, TeeProcs;

type
  TFormAnimatedZoom =
class(TForm)
    Chart1: TChart;
    Panel1: TPanel;
    CheckBox1: TCheckBox;
    CheckBox2: TCheckBox;
    LineSeries1: TLineSeries;
    LineSeries2: TLineSeries;
```

```
(Sender: TObject);
begin
  BitBtn1.Enabled:=False;
  BitBtn2.Enabled:=False;
  Chart1.ZoomPercent(115);
  BitBtn1.Enabled:=True;
  BitBtn2.Enabled:=True;
end;

<-------> procedure
TFormAnimatedZoom.BitBtn2Click
(Sender: TObject);
begin
  BitBtn1.Enabled:=False;
  BitBtn2.Enabled:=False;
  Chart1.ZoomPercent(85);
  BitBtn1.Enabled:=True;
  BitBtn2.Enabled:=True;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

19

{******************************
***************}
{ TeeChart Delphi Component
Library          }
{ Gantt Series Type Demo
}
{ Copyright (c) 1995-2001 by David
Berneda    }
{ All rights reserved                    }
{******************************
***************}
unit gantt;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, Chart, Series,
ExtCtrls, StdCtrls, GanttCh,
  Teengine, Buttons, TeeProcs;

```

```
      Add( tmp, '', clTeeColor );
     end;
   end;
  End;

begin

SpinEdit1.Value:=Chart1.Animated
ZoomSteps;
  RandomValues(LineSeries1);
  RandomValues(LineSeries2);
  RandomValues(LineSeries3);
  RandomValues(LineSeries4);
  RandomValues(LineSeries5);
  RandomValues(PointSeries1);
  Chart1.AnimatedZoom:=False;
  Chart1.ZoomPercent(5);
  Chart1.AnimatedZoom:=True;
end;

<-------> procedure
TFormAnimatedZoom.CheckBox1C
lick(Sender: TObject);
begin

Chart1.AnimatedZoom:=CheckBox1
.checked;
end;

<-------> procedure
TFormAnimatedZoom.CheckBox2C
lick(Sender: TObject);
begin

Chart1.View3D:=CheckBox2.Check
ed;
end;

<-------> procedure
TFormAnimatedZoom.SpinEdit1Ch
ange(Sender: TObject);
begin

Chart1.AnimatedZoomSteps:=SpinE
dit1.Value;
end;

<-------> procedure
TFormAnimatedZoom.BitBtn1Click
```

Use the AddGantt or AddGanttColor methods.
(This demo do not uses this methods)
Example:
GanttSeries1.AddGantt(EncodeDate(1997, 1, 1),
EncodeDate(1997, 1, 31),
0,
'Programming');
Or...
GanttSeries1.AddGanttColor(EncodeDate(1997, 1, 1),
EncodeDate(1997, 1, 31),
0,
'Programming',
clGreen);
Where "0" is the desired vertical position for this bar.
Choose the vertical position you prefer.
To connect gantt bars:
1) Store the "AddGantt" or "AddGanttColor" function return longint:
Var tmp1, tmp2 : Longint;
tmp1:=GanttSeries1.AddGantt(EncodeDate(1997, 1, 1),
EncodeDate(1997, 1, 31),
0,
'Programming');
tmp2:=GanttSeries1.AddGantt(EncodeDate(1997, 4, 1),

type
TGanttForm = class(TForm)
Chart1: TChart;
Panel1: TPanel;
Label1: TLabel;
Shape1: TShape;
BitBtn3: TBitBtn;
GanttSeries1: TGanttSeries;
Memo1: TMemo;
Label2: TLabel;
<-------> procedure FormCreate(Sender: TObject);
<-------> procedure Chart1MouseMove(Sender: TObject; Shift: TShiftState; X,
Y: Integer);
<-------> procedure Chart1GetNextAxisLabel(Sender: TChartAxis;
LabelIndex: Longint; var LabelValue: Double; var Stop: Boolean);
<-------> procedure GanttSeries1Click(Sender: TChartSeries; ValueIndex: Integer;
Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
private
{ Private declarations }
public
{ Public declarations }
end;
implementation
{$R *.dfm}
Uses UDemUtil;
(*
****
HOW TO ADD GANTT BARS MANUALLY ?
****

```
eries1.XValues[tmpTask]);

Label2.Caption:=DateToStr(GanttS
eries1.EndValues[tmpTask]);

Shape1.Brush.Color:=GanttSeries1.
ValueColor[tmpTask];
  end
  else
  begin  { clear label and shape
samples }
    Label1.Caption:='';
    Label2.Caption:='';

Shape1.Brush.Color:=Panel1.Color;
  end;
end;

<-------> procedure
TGanttForm.Chart1GetNextAxisLa
bel(Sender: TChartAxis;
  LabelIndex: Longint; var
LabelValue: Double; var Stop:
Boolean);
var year,month,day:Word;
begin
  { this will set the Bottom Axis
(Dates) Labels to the First
    day of each month. }
  if Sender=Chart1.BottomAxis then
  begin
    { LabelValue has the "candidate"
value where the Axis label will be
painted. }

DecodeDate(LabelValue,year,month,
day);
    { we force that value to be the first
day in month }
    Day:=1;
    Month:=Month+1;
    if Month>12 then
    Begin
      Month:=1;
      Year:=Year+1;
    end;
    { Then we set the preferred Label
value }

                    EncodeDate(
1997, 4, 30 ),
                    0,
                    'Testing' );

   2) Then use the NextTask
property:

    GanttSeries1.NextTask[ tmp1 ] :=
tmp2 ;

    This will draw a line from
'Programming' gantt bar to
'Testing' bar.

    The "ConnectingLinePen"
property is the pen used to draw
lines.

*)

<-------> procedure
TGanttForm.FormCreate(Sender:
TObject);
begin

GanttSeries1.FillSampleValues(Gant
tSeries1.NumSampleValues); { <--
Some random points }
  { change cursor when mouse moves
over Gantt bars... }
  GanttSeries1.Cursor:=crTeeHand;
end;

<-------> procedure
TGanttForm.Chart1MouseMove(Se
nder: TObject; Shift: TShiftState;
  X, Y: Integer);
var tmpTask:Longint;
begin

tmpTask:=GanttSeries1.Clicked(x,y)
;  { <-- if mouse is over a gantt bar...
}
  if tmpTask<>-1 then
  begin { set a sample label and a
sample shape color }

Label1.Caption:=DateToStr(GanttS
```

```
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, Teengine, Series,
ExtCtrls, Chart, StdCtrls,
  Buttons, TeeProcs, TeeFunci;

type
  THighLowForm = class(TForm)
    Chart1: TChart;
    BarSeries1: TBarSeries;
    Panel1: TPanel;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    CheckBox1: TCheckBox;
    Timer1: TTimer;
    CB3D: TCheckBox;
    AverageSeries: TLineSeries;
    HighSeries: TLineSeries;
    LowSeries: TLineSeries;
    CheckBox2: TCheckBox;
    CheckBox3: TCheckBox;
    Memo1: TMemo;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
Chart1AfterDraw(Sender:
TObject);
    <-------> procedure
BitBtn1Click(Sender: TObject);
    <-------> procedure
CheckBox1Click(Sender: TObject);
    <-------> procedure
Timer1Timer(Sender: TObject);
    <-------> procedure
CB3DClick(Sender: TObject);
    <-------> procedure
CheckBox2Click(Sender: TObject);
    <-------> procedure
CheckBox3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    Bar2:TBarSeries;
  end;

implementation

{$R *.dfm}
```

```
LabelValue:=EncodeDate(year,mont
h,day);
    { we want more labels !! }
    Stop:=False;
  end;
end;

<-------> procedure
TGanttForm.GanttSeries1Click(Sen
der: TChartSeries;
  ValueIndex: Integer; Button:
TMouseButton; Shift: TShiftState;
X,
  Y: Integer);
begin
  if Button=mbLeft then
  With Sender do

ValueColor[ValueIndex]:=EditColor
(Self,ValueColor[ValueIndex])
  else
    Chart1.CancelMouse:=False;
    { setting CancelMouse to True
notifies TeeChart engine to STOP
      handling the mouse click and
NOT start zoom or scroll. }
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
20
{******************************
***************}
{ TeeChart Delphi Component
Library          }
{ High-Mean-Low Form Demo
}
{ Copyright (c) 1996-2001 by David
Berneda    }
{ All rights reserved                }
{******************************
***************}
unit uhighlo;

interface

uses
  SysUtils, WinTypes, WinProcs,
```

;
Font.Color:=HighSeries.SeriesColor ;
TextOut(ChartXCenter,
HighSeries.CalcYPos(0),
HighSeries.Name+' '+HighSeries.ValueMarkText[0]);
Font.Color:=LowSeries.SeriesColor;
TextOut(ChartXCenter,
LowSeries.CalcYPos(0),
LowSeries.Name+' '+LowSeries.ValueMarkText[0]);
end;
end;
<-------> procedure THighLowForm.BitBtn1Click(Sender: TObject);
begin
Chart1.PrintLandscape;
end;
<-------> procedure THighLowForm.CheckBox1Click(Sender: TObject);
begin
Timer1.Enabled:=CheckBox1.Checked;
end;
<-------> procedure THighLowForm.Timer1Timer(Sender: TObject);
var tmp:Longint;
begin
Timer1.Enabled:=False;
tmp:=Random(BarSeries1.Count);
BarSeries1.YValue[tmp]:=BarSeries1.YValue[tmp]+Random(50)-25;
if random(100)<8 then
BarSeries1.BarStyle:=TBarStyle(Random(1+Ord(High(TBarStyle))));

<-------> procedure THighLowForm.FormCreate(Sender: TObject);
begin
TeeEraseBack:=False; { try win95 + plus ! + drag window + resizing ! }
Bar2:=nil;
Chart1.View3D:=CB3D.Checked;
Chart1.Chart3DPercent:=35;
BarSeries1.Fillsamplevalues(6);
BarSeries1.Dark3D:=Chart1.IsScreenHighColor;
Chart1.RightAxis.Minimum:=0;
Chart1.RightAxis.AutomaticMaximum:=True;
BarSeries1.RefreshSeries;
end;
<-------> procedure THighLowForm.Chart1AfterDraw(Sender: TObject);
begin
if not CheckBox2.Checked then { only with one bar, much pretty... }
With Chart1,Canvas do
begin
Brush.Style:=bsClear;
Font.PixelsPerInch:=Screen.PixelsPerInch;
Font.Size:=12;
Font.Style:=[fsBold,fsItalic];
Font.Color:=AverageSeries.SeriesColor;
TextOut(ChartXCenter,
AverageSeries.CalcYPos(0),
AverageSeries.Name+' '+AverageSeries.ValueMarkText[0])

```
LowSeries.DataSources.Add(Bar2);
Bar2.AddLinkedSeries(LowSeries);
  end
  else
  begin
   { remove the second bar series we created before... }
     Bar2.Free;
     Bar2:=nil;
  end;
{ and finally refresh the statistical series to view results... }
  BarSeries1.RefreshSeries;
end;

<-------> procedure THighLowForm.CheckBox3Click(Sender: TObject);
begin
BarSeries1.Active:=CheckBox3.Checked;
  if Assigned(Bar2) then
Bar2.Active:=CheckBox3.Checked;
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
21
{*********************************************}
{ TeeChart Delphi Component Library          }
{ Keyboard Scrolling Demo
}
{ Copyright (c) 1996-2001 by David Berneda    }
{ All rights reserved                          }
{*********************************************}
unit ukeyboa;

interface

uses
  SysUtils, WinTypes, WinProcs,
```

```

  { Randomly change from 3D to 2D
}
  if random(100)<2 then CB3D.Checked:=not CB3D.Checked;
  Timer1.Enabled:=True;
end;

<-------> procedure THighLowForm.CB3DClick(Sender : TObject);
begin
  Chart1.View3d:=CB3D.Checked;
end;

<-------> procedure THighLowForm.CheckBox2Click(Sender: TObject);
begin
  if CheckBox2.Checked then
  begin
  { create a new TBarSeries, and add data }
    Bar2:=TBarSeries.Create(Self);
    Bar2.ParentChart:=Chart1;

    Bar2.BarStyle:=bsInvPyramid;

  { fill the second series with same number of random values than
    the first bar series }
    Bar2.FillSampleValues( BarSeries1.Count );

  { then add this new Series as datasource for statistics... }
AverageSeries.DataSources.Add(Bar2);
Bar2.AddLinkedSeries(AverageSeries);

HighSeries.DataSources.Add(Bar2);
Bar2.AddLinkedSeries(HighSeries);
```

```
Messages, Classes, Graphics, Controls,
  Forms, Dialogs, Teengine, Series, ExtCtrls, Chart, StdCtrls, Buttons,
  TeeProcs;

type
  TKeyboardForm = class(TForm)
    Chart1: TChart;
    LineSeries1: TLineSeries;
    Panel1: TPanel;
    BitBtn1: TBitBtn;
    InvertScroll: TCheckBox;
    CheckLimits: TCheckBox;
    Memo1: TMemo;
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    <-------> procedure CheckLimitsClick(Sender: TObject);
    <-------> procedure InvertScrollClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}

<-------> procedure TKeyboardForm.FormCreate(Sender: TObject);
begin
  LineSeries1.FillSampleValues(500);
  AnimatedZoomFactor:=4;
end;

<-------> procedure TKeyboardForm.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
Var XDelta,YDelta,
```

```
   XRange,YRange:Double;
begin
  { initialize some temporary variables... }
  XDelta:=0;
  YDelta:=0;
  With LineSeries1.GetHorizAxis do XRange:=Maximum-Minimum;
  With LineSeries1.GetVertAxis  do YRange:=Maximum-Minimum;
  { handle keyboard !!! }
  if ssShift in Shift then
  begin
    Case key of
      VK_LEFT,VK_UP    : Chart1.ZoomPercent( 110 );
      VK_RIGHT,VK_DOWN : Chart1.ZoomPercent( 90 );
    end;
    exit;
  end
  else
  Case key of
    VK_LEFT  : XDelta:=-XRange/100;
    VK_RIGHT : XDelta:= XRange/100;
    VK_UP    : YDelta:= YRange/100;
    VK_DOWN  : YDelta:=-YRange/100;
    vk_Next  : YDelta:=-YRange/10;
    vk_Prior : YDelta:= YRange/10;
    VK_SPACE : Begin Chart1.UndoZoom; Exit; End;  { <-- reset scrolling }
  end;
  { just to make this example a little better... }
  if not InvertScroll.Checked then
  begin
    XDelta:=-XDelta;
    YDelta:=-YDelta;
  end;
  { apply scrolling !!! }
  With Chart1 do
  Begin
    LeftAxis.Scroll(YDelta,CheckLimits.Checked);
```

```
Paintbox component.
}
uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, ExtCtrls, Chart, Series, StdCtrls, Teengine, Buttons,
  TeeProcs;

type
  TLegendForm = class(TForm)
    Chart1: TChart;
    LineSeries1: TLineSeries;
    LineSeries2: TLineSeries;
    PaintBox1: TPaintBox;
    Panel1: TPanel;
    BitBtn3: TBitBtn;
    Label1: TLabel;
    Memo1: TMemo;
    <-------> procedure LineSeries2AfterDrawValues(Sender: TObject);
    <-------> procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  LegendForm: TLegendForm;

implementation

{$R *.dfm}

<-------> procedure TLegendForm.LineSeries2AfterDrawValues(Sender: TObject);
var t:Longint;
begin
  With Paintbox1.Canvas do  { we'll draw over PaintBox1 }
  Begin
    for t:=0 to Chart1.SeriesCount-1 do  { for each Series in Chart... }
    Begin
```

```

RightAxis.Scroll(YDelta,CheckLimits.Checked);

BottomAxis.Scroll(XDelta,CheckLimits.Checked);

TopAxis.Scroll(XDelta,CheckLimits.Checked);
    SetFocus;
  End;
end;

<-------> procedure TKeyboardForm.CheckLimitsClick(Sender: TObject);
begin
  ShowMessage('Please zoom before scrolling.');
  Chart1.SetFocus;
end;

<-------> procedure TKeyboardForm.InvertScrollClick(Sender: TObject);
begin
  Chart1.SetFocus;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\\\\
22
{*******************************************}
{    TeeChart. Legend Example
}
{ Copyright (c) 1995-2001 by David Berneda }
{    All Rights Reserved                }
{*******************************************}
unit ulegend;

interface
{ This form shows how to create a new Chart.Legend.
  We'll draw the Series Titles onto a
```

```
Messages, Classes, Graphics, Controls,
  Forms, Dialogs, ExtCtrls, Chart, Series, StdCtrls, Teengine, Buttons,
  TeeProcs;

type
  TLegendXYForm = class(TForm)
    Chart1: TChart;
    PieSeries1: TPieSeries;
    Panel1: TPanel;
    RadioGroup1: TRadioGroup;
    BitBtn3: TBitBtn;
    Memo1: TMemo;
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure Chart1GetLegendRect(Sender: TCustomChart; var Rect: TRect);
    <-------> procedure Chart1GetLegendPos(Sender: TCustomChart; Index: Longint;
      var X, Y, XColor: Longint);
    <-------> procedure RadioGroup1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    DefaultLegend:Boolean;
  end;

var
  LegendXYForm: TLegendXYForm;

implementation

{$R *.dfm}

<-------> procedure TLegendXYForm.FormCreate(Sender: TObject);
begin
  DefaultLegend:=False;            { <-- only used in this example }
  PieSeries1.FillSampleValues(10);   { <-- some random pie sectors }
```

```
Font.Color:=Chart1[t].SeriesColor; { set font color }
     { draw the customized Series Title }
     TextOut(40,20+16*t,'This is a long Series title: '+Chart1.SeriesTitleLegend(t));
   end;
  End;
end;

<-------> procedure TLegendForm.FormCreate(Sender: TObject);
begin
  LineSeries1.FillSampleValues(50); { random values }
  LineSeries2.FillSampleValues(50);
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

24

```
{*********************************************}
{ TeeChart Delphi Component Library          }
{ Custom Legend Size and Position Demo       }
{ Copyright (c) 1995-2001 by David Berneda    }
{ All rights reserved                         }
{*********************************************}
unit uylegend;

interface
{ This form shows a customized Legend.
  The Chart.OnGetLegendRect and Chart.OnGetLegendPos events are used to
  change the default legend size and the default legend text positions.
}
uses
  SysUtils, WinTypes, WinProcs,
```

```
k(Sender: TObject);
begin
  { Get the RadioGroup selection and force the chart to repaint }

DefaultLegend:=RadioGroup1.ItemIndex=0;
  Chart1.Repaint;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
25
{*********************************************}
{ TeeChart Delphi Component Library          }
{ Linked Tables Chart Demo
}
{ Copyright (c) 1995-2001 by David Berneda    }
{ All rights reserved                          }
{*********************************************}
unit linked;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, Buttons, ExtCtrls, Grids, DBGrids, DB,
  DBTables, Chart, Series, DbChart, DBCtrls, Teengine, TeeProcs;

type
  TLinkedTablesForm = class(TForm)
    DBChart1: TDBChart;
    DataSource1: TDataSource;
    DBGrid1: TDBGrid;
    Panel1: TPanel;
    Panel2: TPanel;
    Table1: TTable;
    Table2: TTable;
    DBNavigator1: TDBNavigator;
```

```
end;

<-------> procedure TLegendXYForm.Chart1GetLegendRect(Sender: TCustomChart;
  var Rect: TRect);
begin
  if not DefaultLegend then  { <-- if we want to customize legend... }
  Begin
    { This changes the Legend Rectangle dimensions }

Rect.Bottom:=Rect.Top+PieSeries1.Count*15;  { <-- Calc Legend Height }
    Rect.Left:=Rect.Left-120;
{ <-- Bigger Legend Width }
  end;
end;

<-------> procedure TLegendXYForm.Chart1GetLegendPos(Sender: TCustomChart; Index: Longint;
  var X, Y, XColor: Longint);
begin
  if not DefaultLegend then
  Begin
    { Calculate the X Y coordinates for each Legend Text }

x:=Chart1.Legend.RectLegend.Left;
    x:=x + (Index div (PieSeries1.Count div 2))*100;


y:=Chart1.Legend.RectLegend.Top;
    y:=y +   (Index mod (PieSeries1.Count div 2))*30;

    if (Index mod 2)=1 then X:=X+20;
    x:=x+20;
    XColor:=X-15;
  end;
end;

<-------> procedure TLegendXYForm.RadioGroup1Clic
```

```
Berneda    }
{ All rights reserved                    }
{*****************************
***************}
unit loglab;
{$P-}

interface

{ This form shows how custom Axis
labels can be specified }
{ The Chart.OnGetNextAxisLabel
event is used to supply the axis with
  custom label positions and values. }

uses
  WinProcs, WinTypes, Messages,
SysUtils, Classes, Graphics,
Controls, Forms, Dialogs,
  ExtCtrls, TeeProcs, TeEngine,
Chart, Series, StdCtrls, Buttons;

type
  TLogLabelsForm = class(TForm)
    Chart1: TChart;
    Series1: TFastLineSeries;
    Panel1: TPanel;
    BitBtn2: TBitBtn;
    Memo1: TMemo;
    <-------> procedure
FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}

<-------> procedure
TLogLabelsForm.FormCreate(Send
er: TObject);
begin
  { Axis settings }

Chart1.BottomAxis.Logarithmic:=T
rue;
```

```
    DataSource2: TDataSource;
    DBGrid2: TDBGrid;
    Table2OrderNo: TFloatField;
    Table2CustNo: TFloatField;
    Table2SaleDate: TDateTimeField;
    Table2AmountPaid:
TCurrencyField;
    Table1CustNo: TFloatField;
    Table1Company: TStringField;
    Table1City: TStringField;
    Table1State: TStringField;
    Table1Country: TStringField;
    AreaSeries1: TAreaSeries;
    BitBtn1: TBitBtn;
    <-------> procedure
DataSource1DataChange(Sender:
TObject; Field: TField);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}

<-------> procedure
TLinkedTablesForm.DataSource1D
ataChange(Sender: TObject;
  Field: TField);
begin  { force dbchart to refresh ! }

DBChart1.CheckDataSource(AreaS
eries1);
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

26
{*****************************
***************}
{ TeeChart Delphi Component
Library          }
{ Logarithmic Labels Demo
}
{ Copyright (c) 1996-2001 by David
```

```
  Forms, Dialogs, Teengine, Series, ExtCtrls, Chart, StdCtrls,
  Buttons, TeeProcs;

type
  TMetafileForm = class(TForm)
    Chart1: TChart;
    Image1: TImage;
    BarSeries1: TBarSeries;
    Panel1: TPanel;
    Panel2: TPanel;
    SaveDialog1: TSaveDialog;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    BitBtn4: TBitBtn;
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure BitBtn3Click(Sender: TObject);
    <-------> procedure BitBtn2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}
uses Clipbrd;

<-------> procedure TMetafileForm.FormCreate(Sender: TObject);
begin
  BarSeries1.FillSampleValues(8);  { <-- some sample random bars }
end;

{ This code copies Chart contents onto Windows Clipboard in Metafile Format }
<-------> procedure TMetafileForm.BitBtn3Click(Sender : TObject);
begin
{    TeeClipWhenMetafiling:=True; { <--- FORCE CLIPPING WITH
```

```
Chart1.BottomAxis.TickOnLabelsOnly:=True;
  Chart1.BottomAxis.SetMinMax( 10.0, 1000);

Chart1.LeftAxis.Logarithmic:=True;

Chart1.LeftAxis.TickOnLabelsOnly:=True;
  Chart1.LeftAxis.SetMinMax( 10.0, 1000);

  { adding XY values to Series1 }
  Series1.XValues.DateTime:=False;
  Series1.AddXY( 100, 100, '', clTeeColor );
  Series1.AddXY( 500, 200, '', clTeeColor );
  Series1.AddXY( 800, 300, '', clTeeColor );
  Series1.AddXY( 200, 200, '', clTeeColor );
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
27
{*********************************************}
{ TeeChart Delphi Component Library          }
{ Metafile *.WMF Demo                         }
{ Copyright (c) 1995-2001 by David Berneda    }
{ All rights reserved                         }
{*********************************************}
unit umetafil;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
```

```
    Rect( 0,0, round(21{cm}*37.8),
round(10{cm}*37.8)));
*)
{  ( this equals to 96 * 21 / 2.54 , 96 *
10 /2.54 )  }

   { now it's loaded HERE ! }

Image1.Picture.LoadFromFile(Save
Dialog1.FileName);
   Image1.Refresh;
  end;
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
28
{******************************
************}
{   TeeChart. Mini-Charts Example
}
{ Copyright (c) 1995-2001 by David
Berneda }
{   All Rights Reserved              }
{******************************
************}
unit umain;

interface

{ This project show some very small
mini-charts and some animation }
{ That mini-charts could be useful as
small data-monitors in your forms,
  or reports }
uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, Chart, Series,
ExtCtrls, Teengine, TeeProcs;

type
  TMiniForm = class(TForm)
   LineSeries1: TLineSeries;
   Panel1: TPanel;
   Panel2: TPanel;
   Chart1: TChart;
   Chart2: TChart;
```

```
METAFILES }
{   CLIPPING WORKS FINE BUT
DO NOT ALLOW MOVEABLE
OR RESIZEABLE METAFILES }


Chart1.CopyToClipboardMetafile(T
rue);  { <--- Enhanced Metafile =
True }

  ShowMessage('Chart1 is now at
Windows Clipboard in Metafile
format.'+#13+
              'and will now be pasted
HERE !');
{ Now PASTE! }
  Image1.Picture.Assign(ClipBoard);
  Image1.Refresh;
end;

{ This button asks a filename and
saves the Chart }
<-------> procedure
TMetafileForm.BitBtn2Click(Sender
: TObject);
begin
  if SaveDialog1.Execute then  { <--
ask for a filename first }
  begin
   { SAVE IT !! }

{   CLIPPING WORKS FINE BUT
DO NOT ALLOW MOVEABLE
OR RESIZEABLE METAFILES }
{   TO FORCE CLIPPING WITH
METAFILES UNCOMMENT THIS
LINE: }
{   TeeClipWhenMetafiling:=True;
}


Chart1.SaveToMetafile(SaveDialog1
.FileName);

   { THIS METHOD CAN BE USED
TOO: }
(*
      Chart1.SaveToMetafileRect(
SaveDialog1.FileName,
```

```
AreaSeries1.FillSampleValues(NumPoints);
LineSeries2.FillSampleValues(NumPoints);
  BarSeries1.FillSampleValues(6);
LineSeries3.FillSampleValues(NumPoints);
end;

<-------> procedure TMiniForm.FormResize(Sender: TObject);
begin
  { Equally resize the panels to center charts }
  Chart2.Height:=ClientHeight div 2;
  Chart3.Height:=ClientHeight div 2;
  Panel1.Width:=ClientWidth div 2;
end;

<-------> procedure TMiniForm.Timer1Timer(Sender: TObject);

  { This <-------> procedure changes the Series values every second }
  <-------> procedure RefreshMonitorChart(Chart:TChart; Var PosChart:Longint);
  var t:Longint;
      LastValueWas:Double;
  Begin
    Inc(PosChart);
    if PosChart >= NumPoints then PosChart:=0;
    for t:=0 to Chart.SeriesCount-1 do
    Begin
      if PosChart=0 then
      Begin
        With Chart do { reset scales at the end of monitoring. }
        Begin
          LeftAxis.Automatic:=True;
LeftAxis.SetMinMax(MinYValue(LeftAxis),MaxYValue(LeftAxis));
        end;
    Chart3: TChart;
    Chart4: TChart;
    PieSeries1: TPieSeries;
    AreaSeries1: TAreaSeries;
    LineSeries2: TLineSeries;
    BarSeries1: TBarSeries;
    LineSeries3: TLineSeries;
    Timer1: TTimer;
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure FormResize(Sender: TObject);
    <-------> procedure Timer1Timer(Sender: TObject);
    <-------> procedure LineSeries2AfterDrawValues(Sender: TObject);
    <-------> procedure LineSeries3AfterDrawValues(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    PosChart1,PosChart4:Longint;  { used to draw vertical lines over charts }
  end;

implementation

{$R *.dfm}

Const NumPoints=30;

<-------> procedure TMiniForm.FormCreate(Sender: TObject);
begin
  PosChart1:=-1;                 { starting position of vertical divider }
  PosChart4:=NumPoints div 2;

  { Generate some random points ... }
LineSeries1.FillSampleValues(NumPoints);
  PieSeries1.FillSampleValues(8);

```

```
  if PosChart1>=0 then
  With Chart1,Canvas do
  Begin
    Pen.Color:=clRed;
    DoVertLine(
Series[0].CalcXPos(PosChart1),  { x
}
              ChartRect.Top+1,
{ initial Y }
              ChartRect.Bottom-1
{ ending Y }
            );
  end;
end;

<-------> procedure
TMiniForm.LineSeries3AfterDrawV
alues(Sender: TObject);
begin
  { this event draws the blue divider
in Chart4 }
  if PosChart4>=0 then
  With Chart4,Canvas do
  Begin
    Pen.Color:=clBlue;
    DoVertLine(
Series[0].CalcXPos(PosChart4),  { x
}
              ChartRect.Top+1,
{ initial Y }
              ChartRect.Bottom-1
{ ending Y }
             );
  end;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
29

{******************************
**************}
{ TeeChart Delphi Component
Library          }
{ Overlayed Bars Demo
}
{ Copyright (c) 1995-2001 by David
Berneda    }
```

```

LastValueWas:=Chart.Series[t].YVa
lues.Last;
     end
     else
LastValueWas:=Chart.Series[t].YVa
lue[PosChart-1];

     { change the value for a new
random one }

Chart.Series[t].YValue[PosChart]:=
LastValueWas+Random(ChartSam
plesMax)-

(ChartSamplesMax div 2);
   end;
  end;

var tmpPos:Longint;
begin

RefreshMonitorChart(Chart1,PosC
hart1);      { refresh chart1  }


RefreshMonitorChart(Chart4,PosC
hart4);      { refresh chart4  }

  With PieSeries1 do
RotationAngle:=(RotationAngle+1)
mod 359;  { rotate pie }

  { change Bar Series values }
  With BarSeries1 do
  Begin
    tmpPos:=Random(Count);

YValue[tmpPos]:=YValue[tmpPos]*
(80.0+Random(40))/100.0;
  end;
end;

<-------> procedure
TMiniForm.LineSeries2AfterDrawV
alues(Sender: TObject);
begin
  { this event draws the red divider in
Chart1 }
```

```
{ All rights reserved                    }
{******************************************************}
unit uoverbar;

interface

{ This form shows 3 bar series in a overlayed layout.
  Each Bar Series has a different BarWidthPercent.
  The order Series are drawn is the most important thing. See below.
}
uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, Chart, Series, ExtCtrls, Teengine, StdCtrls, Spin,
  Buttons, TeeProcs;

type
  TOverBarForm = class(TForm)
    Chart1: TChart;
    BarSeries1: TBarSeries;
    BarSeries2: TBarSeries;
    BarSeries3: TBarSeries;
    Panel1: TPanel;
    Memo1: TMemo;
    BitBtn3: TBitBtn;
    Label1: TLabel;
    SpinEdit1: TSpinEdit;
    Label2: TLabel;
    SpinEdit2: TSpinEdit;
    CBPatterns: TCheckBox;
    <-------> procedure FormCreate(Sender: TObject);
    <-------> procedure BarSeries2Click(Sender: TChartSeries; ValueIndex: Integer;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    <-------> procedure BarSeries1Click(Sender: TChartSeries; ValueIndex: Integer;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    <-------> procedure SpinEdit1Change(Sender: TObject);
    <-------> procedure SpinEdit2Change(Sender: TObject);
    <-------> procedure CBPatternsClick(Sender: TObject);
    <-------> procedure BarSeries3Click(Sender: TChartSeries; ValueIndex: Integer;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
  private
    { Private declarations }
  public
    { Public declarations }
    <-------> procedure ChangeOverBars;
  end;

var
  OverBarForm: TOverBarForm;

implementation

{$R *.dfm}

<-------> procedure TOverBarForm.FormCreate(Sender : TObject);
var t:Longint;
begin
  { lets fill the 3 Bar Series with some random data }
  { Series3 has the biggest values, while Series1 has the smaller values }
  With BarSeries3 do
  begin
    Clear;
    for t:=1 to 10 do Add( 100+Random(30),'', clTeeColor);
  end;
  With BarSeries2 do
  begin
    Clear;
    for t:=1 to 10 do Add( 50+Random(10), '', clTeeColor);
  end;
  With BarSeries1 do
  begin
```

```
TOverBarForm.SpinEdit1Change(S
ender: TObject);
begin
  ChangeOverBars;
end;

<-------> procedure
TOverBarForm.SpinEdit2Change(S
ender: TObject);
begin { increase / decrease the 3
Series OffsetPercent property }
{ This will make partially overlay }

BarSeries3.OffsetPercent:=SpinEdit
2.Value;

BarSeries1.OffsetPercent:=SpinEdit
2.Value-
Round(SpinEdit2.Value*55.0/100.0);

BarSeries2.OffsetPercent:=SpinEdit
2.Value-
Round(SpinEdit2.Value*35.0/100.0);
end;

<-------> procedure
TOverBarForm.CBPatternsClick(Se
nder: TObject);
begin
  if CBPatterns.Checked then
  begin

BarSeries1.BarBrush.Style:=bsFDia
gonal;

BarSeries2.BarBrush.Style:=bsBDia
gonal;

BarSeries3.BarBrush.Style:=bsDiag
Cross;
    Chart1.BackColor:=clWhite;
  end
  else
  begin

BarSeries1.BarBrush.Style:=bsSolid
;

BarSeries2.BarBrush.Style:=bsSolid
```

```
   Clear;
   for t:=1 to 10 do Add(
20+Random(10), '', clTeeColor);
  end;
  ChangeOverBars;
end;

<-------> procedure
TOverBarForm.BarSeries2Click(Se
nder: TChartSeries; ValueIndex:
Integer;
  Button: TMouseButton; Shift:
TShiftState; X, Y: Integer);
begin
  Showmessage('You clicked the Red
Bar at point #:
'+inttostr(Valueindex));
end;

<-------> procedure
TOverBarForm.BarSeries1Click(Se
nder: TChartSeries; ValueIndex:
Integer;
  Button: TMouseButton; Shift:
TShiftState; X, Y: Integer);
begin
  Showmessage('You clicked the
Green Bar at point #:
'+inttostr(Valueindex));
end;

<-------> procedure
TOverBarForm.ChangeOverBars;
begin { compress / expand the
overlayed space between bars }

BarSeries3.BarWidthPercent:=Spin
Edit1.Value;

BarSeries1.BarWidthPercent:=Spin
Edit1.Value-
Round(SpinEdit1.Value*55.0/100.0);

BarSeries2.BarWidthPercent:=Spin
Edit1.Value-
Round(SpinEdit1.Value*35.0/100.0);
end;

<-------> procedure
```

```
returns the current Chart Page.
   Chart.NumPages : Longint
Returns the Total Number of Pages.
   method Chart.NextPage ;     Goes
to next page.
   method Chart.PreviousPage ;
Goes to previous page.
}
uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, Chart, Series,
ExtCtrls, StdCtrls, Spin, BubbleCh,
Buttons,
  Teengine, TeeProcs;

type
  TPagesForm = class(TForm)
    Chart1: TChart;
    LineSeries1: TLineSeries;
    BubbleSeries1: TBubbleSeries;
    Panel1: TPanel;
    SpinEdit1: TSpinEdit;
    Label2: TLabel;
    Label1: TLabel;
    ButtonPrevious: TBitBtn;
    ButtonNext: TBitBtn;
    BLastPage: TBitBtn;
    BFirstPage: TBitBtn;
    CheckBox1: TCheckBox;
    BitBtn3: TBitBtn;
    Memo1: TMemo;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
SpinEdit1Change(Sender: TObject);
    <-------> procedure
Chart1PageChange(Sender:
TObject);
    <-------> procedure
ButtonPreviousClick(Sender:
TObject);
    <-------> procedure
ButtonNextClick(Sender: TObject);
    <-------> procedure
CheckBox1Click(Sender: TObject);
    <-------> procedure
BFirstPageClick(Sender: TObject);
```

```
;

BarSeries3.BarBrush.Style:=bsSolid
;
   Chart1.BackColor:=clTeeColor;
  end;
end;

<-------> procedure
TOverBarForm.BarSeries3Click(Se
nder: TChartSeries;
  ValueIndex: Integer; Button:
TMouseButton; Shift: TShiftState;
X,
  Y: Integer);
begin
   Showmessage('You clicked the
Blue Bar at point #:
'+inttostr(Valueindex));
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
30
{*******************************
************}
{    TeeChart. TChart Component
}
{ Copyright (c) 1995-2001 by David
Berneda }
{    All Rights Reserved               }
{*******************************
************}
unit upages;

interface
{ This form demonstrates how
Charts can be divided in several
Pages.
  The main Chart property is
"MaxPointsPerPage".
  TeeChart automatically calculates
the total number of pages based on
that
  property.
  The other Paging properties and
methods are:

   Chart.Page     : Longint    Sets and
```

```
  Chart1.UndoZoom;
  { show the current page number and the total number of pages }
  { (like a report) }
Label1.Caption:=IntToStr(Chart1.Page)+'/'+IntToStr(Chart1.NumPages);
  { enable or disable buttons }
ButtonPrevious.Enabled:=Chart1.Page > 1;
  ButtonNext.Enabled:=Chart1.Page < Chart1.NumPages;
BLastPage.Enabled:=ButtonNext.Enabled;
BFirstPage.Enabled:=ButtonPrevious.Enabled;
end;

<-------> procedure TPagesForm.ButtonPreviousClick(Sender: TObject);
begin
  Chart1.PreviousPage; { <-- goto previous chart page }
end;

<-------> procedure TPagesForm.ButtonNextClick(Sender: TObject);
begin
  Chart1.NextPage;  { <-- goto next chart page }
end;

<-------> procedure TPagesForm.CheckBox1Click(Sender: TObject);
begin
Chart1.ScaleLastPage:=CheckBox1.Checked;  { <-- only for last page }
end;

<-------> procedure TPagesForm.BFirstPageClick(Sende
```

```
  <-------> procedure BLastPageClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  PagesForm: TPagesForm;

implementation

{$R *.dfm}

<-------> procedure TPagesForm.FormCreate(Sender: TObject);
begin
  LineSeries1.FillSampleValues(100); { <-- some random points }
BubbleSeries1.FillSampleValues(100);
  SpinEdit1.Value:=18;        { <-- max number of points per page }
  Chart1PageChange(Chart1);    { <-- repaint page / number of pages }
end;

<-------> procedure TPagesForm.SpinEdit1Change(Sender: TObject);
begin
  { change the max number of points per page }
  { a value of Zero means "No pages. Show all points" }
Chart1.MaxPointsPerPage:=SpinEdit1.Value;
  Chart1PageChange(Chart1); { <-- repaint page / number of pages }
end;

<-------> procedure TPagesForm.Chart1PageChange(Sender: TObject);
begin
```

```
   BitBtn3: TBitBtn;
   CheckBox4: TCheckBox;
   Shape1: TShape;
   RadioGroup1: TRadioGroup;
   CheckBox5: TCheckBox;
   CheckBox6: TCheckBox;
   RadioGroup2: TRadioGroup;
   <-------> procedure
FormCreate(Sender: TObject);
   <-------> procedure
CheckBox1Click(Sender: TObject);
   <-------> procedure
Timer1Timer(Sender: TObject);
   <-------> procedure
CheckBox2Click(Sender: TObject);
   <-------> procedure
CheckBox3Click(Sender: TObject);
   <-------> procedure
PieSeries1Click(Sender:
TChartSeries; ValueIndex: Integer;
     Button: TMouseButton; Shift:
TShiftState; X, Y: Integer);
   <-------> procedure
CheckBox4Click(Sender: TObject);
   <-------> procedure
Chart1MouseMove(Sender:
TObject; Shift: TShiftState; X,
     Y: Integer);
   <-------> procedure
RadioGroup1Click(Sender:
TObject);
   <-------> procedure
CheckBox5Click(Sender: TObject);
   <-------> procedure
CheckBox6Click(Sender: TObject);
  private
   { Private declarations }
  public
   { Public declarations }
   tmpX,tmpY:Longint;
  end;

implementation

{$R *.dfm}
Uses UDemUtil;

<-------> procedure
TPieForm.FormCreate(Sender:
r: TObject);
begin
  Chart1.Page:=1;  { go to first page }
end;

<-------> procedure
TPagesForm.BLastPageClick(Sende
r: TObject);
begin
  Chart1.Page:=Chart1.NumPages;
{ go to last page }
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\
31
{*******************************
**************}
{ TeeChart Delphi Component
Library          }
{ Pie Series Type Demo
}
{ Copyright (c) 1995-2001 by David
Berneda    }
{ All rights reserved                  }
{*******************************
**************}
unit pie;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, Chart, Series,
ExtCtrls, StdCtrls, Teengine,
  Buttons, teeprocs;

type
  TPieForm = class(TForm)
    Chart1: TChart;
    Panel1: TPanel;
    CheckBox1: TCheckBox;
    Timer1: TTimer;
    PieSeries1: TPieSeries;
    CheckBox2: TCheckBox;
    CheckBox3: TCheckBox;
```

```
XRadius+tmpX>Chart1.ChartWidt
h div 3 then tmpX:=-tmpX;
   end
   else if XRadius+tmpX<30 then
tmpX:=-tmpX;

CustomXRadius:=XRadius+tmpX;

   if tmpY>0 then
   Begin
     if
YRadius+tmpY>Chart1.ChartHeigh
t div 3 then tmpY:=-tmpY;
   end
   else
     if YRadius+tmpY<30 then
tmpY:=-tmpY;

CustomYRadius:=YRadius+tmpY;
  end;

{ Change Marks.Font.Color
randomly }
{ if Random(100)<4 then
  With PieSeries1.Marks do
  Begin
   Repeat

BackColor:=ColorPalette[1+Rando
m(MaxDefaultColors)];

Font.Color:=ColorPalette[1+Rando
m(MaxDefaultColors)];
   Until BackColor<>Font.Color;

Arrow.Color:=ColorPalette[1+Rand
om(MaxDefaultColors)];
  end;

  { Change Legend positioning
Randomly }
  if Random(100)<4 then

RadioGroup1.ItemIndex:=Random(
RadioGroup1.Items.Count);
end;

<-------> procedure
TPieForm.CheckBox2Click(Sender:
TObject);
begin
  tmpX:=-4;
  tmpY:=-3;
  PieSeries1.FillSampleValues(8); {
<-- Some random points }

Chart1.Gradient.Visible:=Chart1.Is
ScreenHighColor;
  if Chart1.Gradient.Visible then
     Chart1.Title.Font.Color:=clWhite
  else
     Chart1.Title.Font.Color:=clBlue;

CheckBox6.Checked:=Chart1.Gradi
ent.Visible;
end;

<-------> procedure
TPieForm.CheckBox1Click(Sender:
TObject);
begin

Timer1.Enabled:=CheckBox1.Check
ed; { <-- animation on/off }
  CheckBox3.Enabled:=not
Timer1.Enabled;
  if not Timer1.Enabled then
  Begin
    PieSeries1.CustomXRadius:=0;
    PieSeries1.CustomYRadius:=0;
  end;
end;

<-------> procedure
TPieForm.Timer1Timer(Sender:
TObject);
begin
{ increase pie rotation angle to get
animation }
  PieSeries1.Rotate(5);

{ change X and Y radius to get
animation }
  With PieSeries1 do
  Begin
    if tmpX>0 then
    Begin
      if
```

```
begin
  tmp:=PieSeries1.Clicked(x,y);
  if tmp=-1 then
Shape1.Visible:=False
  else
  begin
    Shape1.Visible:=True;

Shape1.Brush.Color:=PieSeries1.Val
ueColor[tmp];
  end;
end;

<-------> procedure
TPieForm.RadioGroup1Click(Sende
r: TObject);
begin

Chart1.Legend.Alignment:=TLegen
dAlignment(RadioGroup1.ItemInde
x);
end;

<-------> procedure
TPieForm.CheckBox5Click(Sender:
TObject);
begin

PieSeries1.Marks.Visible:=CheckBo
x5.Checked;
end;

<-------> procedure
TPieForm.CheckBox6Click(Sender:
TObject);
begin

Chart1.Gradient.Visible:=CheckBox
6.Checked;
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
33
{*******************************
***************}
{ TeeChart Delphi Component
Library          }
```

```
TObject);
begin

Chart1.View3d:=CheckBox2.Checke
d; { <-- 3d on / off }
end;

<-------> procedure
TPieForm.CheckBox3Click(Sender:
TObject);
begin

PieSeries1.Circled:=CheckBox3.Che
cked; { <-- circled / elliptic pie }
end;

<-------> procedure
TPieForm.PieSeries1Click(Sender:
TChartSeries;
  ValueIndex: Integer; Button:
TMouseButton; Shift: TShiftState;
X,
  Y: Integer);
begin
  { On Clicked Pie, let the user
change the clicked pie color }
  With PieSeries1 do

ValueColor[ValueIndex]:=EditColor
(Self,ValueColor[ValueIndex]);
end;

<-------> procedure
TPieForm.CheckBox4Click(Sender:
TObject);
begin

PieSeries1.UsePatterns:=CheckBox4
.Checked;
  if PieSeries1.UsePatterns then
PieSeries1.CircleBackColor:=clWhit
e;
end;

<-------> procedure
TPieForm.Chart1MouseMove(Sende
r: TObject; Shift: TShiftState; X,
  Y: Integer);
var tmp:Longint;
```

```
{ Mixed Text and Chart Print Demo
}
{ Copyright (c) 1995-2001 by David
Berneda    }
{ All rights reserved                }
{*******************************
***************}
unit uprint;

{ This example shows how to print
both Text and Chart in the SAME
PAGE }

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, StdCtrls, Buttons,
Chart, Series, ExtCtrls, Teengine,
  TeeProcs;

type
  TPrintForm = class(TForm)
    Chart1: TChart;
    LineSeries1: TLineSeries;
    BitBtn1: TBitBtn;
    Edit1: TEdit;
    BitBtn2: TBitBtn;
    Label1: TLabel;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
BitBtn1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  PrintForm: TPrintForm;

implementation

{$R *.dfm}
uses printers;

<-------> procedure
TPrintForm.FormCreate(Sender:
TObject);
begin
  LineSeries1.FillSampleValues(30);
{ <-- we need some random values }
end;

<-------> procedure
TPrintForm.BitBtn1Click(Sender:
TObject);
var h,w:longint;
begin
  Screen.Cursor := crHourGlass; { <-
- nice detail }
  try
    Printer.BeginDoc;      { <-- start
printer job }
    try
      { now print some text on
printer.canvas }
      With Printer.Canvas do
      begin
        Font.Name:='Arial';
        Font.Size:=10;           { <-- set
the font size }
        Font.Style:=[];
        TextOut(0,0,Edit1.Text);   { <--
print some text }
      end;

      h:=Printer.PageHeight; { <-- get
page height }
      w:=Printer.PageWidth;  { <-- get
page width }

      { And now print the chart
component... }
      Chart1.PrintPartial(  Rect(  w
div 10,         { <-- left margin }
                          h div 3 ,         {
<-- top margin }
                          w - (w div 10),
{ <-- right margin }
                          h - (h div 10) ));
{ <-- bottom margin }


      { print more text.... }
```

```
Series, ExtCtrls, Chart, TeeProcs;

type
  TScrollBarForm = class(TForm)
    Chart1: TChart;
    LineSeries1: TLineSeries;
    ScrollBar2: TScrollBar;
    ScrollBar1: TScrollBar;
    <-------> procedure
ScrollBar1Change(Sender:
TObject);
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
ScrollBar2Change(Sender:
TObject);
    <-------> procedure
Chart1Scroll(Sender: TObject);
    <-------> procedure
Chart1Zoom(Sender: TObject);
    <-------> procedure
Chart1UndoZoom(Sender:
TObject);
    <-------> procedure
Chart1Resize(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    ChangingBars:Boolean;
    <-------> procedure
CalcScrollBarPos;
  end;

implementation

{$R *.dfm}

<-------> procedure
TScrollBarForm.ScrollBar1Change(
Sender: TObject);
var Difer:Double;
begin
  if not ChangingBars then
  With Chart1.BottomAxis do
  Begin
    Difer:=Maximum-Minimum;

Maximum:=Chart1.MaxXValue(Ch
```

```
      With Printer.Canvas do
      begin
        Font.Name:='Arial';
        Font.Size:=12;          { <-- set
the font size }
        Font.Style:=[fsItalic];
        TextOut(0,60,Edit1.Text+'
...again');   { <-- print some text }
      end;

      Printer.EndDoc; { <-- end job
and print !! }
    except
      on Exception do  { just in case an
error happens... }
      Begin
        Printer.Abort;
        Printer.EndDoc;
        Raise;      { <-- raise up the
exception !!! }
      end;
    end;
  finally
    Screen.Cursor:=crDefault; { <--
restore cursor }
  end;
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\
34
{******************************
***********}
{ TeeChart                        }
{ Scroll Bars Example             }
{ Copyright (c) 1995-2001 by David
Berneda }
{    All Rights Reserved          }
{******************************
***********}
unit uscrollb;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, StdCtrls, Teengine,
```

```
Round(100.0*(BottomAxis.Minimu
m-MinXValue(BottomAxis))/Difer);
  end;
  if LeftAxis.Automatic then
     ScrollBar2.Enabled:=False
  else
  Begin
    ScrollBar2.Enabled:=True;
    Difer:=MaxYValue(LeftAxis)-
MinYValue(LeftAxis);
    if Difer>0 then
    ScrollBar2.Position:=

Round(100.0*(LeftAxis.Minimum-
MinYValue(LeftAxis))/Difer);
  end;
 end;
 ChangingBars:=False;
End;

<-------> procedure
TScrollBarForm.Chart1Scroll(Send
er: TObject);
begin
 CalcScrollBarPos;
end;

<-------> procedure
TScrollBarForm.Chart1Zoom(Send
er: TObject);
begin
 CalcScrollBarPos;
end;

<-------> procedure
TScrollBarForm.Chart1UndoZoom(
Sender: TObject);
begin
 CalcScrollBarPos;
end;

<-------> procedure
TScrollBarForm.Chart1Resize(Send
er: TObject);
begin
 ScrollBar1.Left:=0;
 ScrollBar1.Top:=Chart1.Height-
ScrollBar1.Height;
art1.BottomAxis)-
ScrollBar1.Position*Difer/100.0;
   Minimum:=Maximum-Difer;
 end;
end;

<-------> procedure
TScrollBarForm.FormCreate(Sende
r: TObject);
begin

LineSeries1.FillSampleValues(1000);
 Chart1.ZoomPercent(115);
 CalcScrollBarPos;
end;

<-------> procedure
TScrollBarForm.ScrollBar2Change(
Sender: TObject);
Var Difer:Double;
begin
 if not ChangingBars then
 With Chart1.LeftAxis do
 Begin
   Difer:=Maximum-Minimum;

Minimum:=Chart1.MinYValue(Cha
rt1.LeftAxis)+ScrollBar2.Position*D
ifer/100.0;
   Maximum:=Minimum+Difer;
 end;
end;

<-------> procedure
TScrollBarForm.CalcScrollBarPos;
Var Difer:Double;
Begin
 ChangingBars:=True;
 With Chart1 do
 Begin
  if BottomAxis.Automatic then
     ScrollBar1.Enabled:=False
  else
  Begin
    ScrollBar1.Enabled:=True;
    Difer:=MaxXValue(BottomAxis)-
MinXValue(BottomAxis);
    if Difer>0 then
    ScrollBar1.Position:=
```

<-------> procedure Button1Click(Sender: TObject);
<-------> procedure CBVerticalClick(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
<-------> procedure AddPoint(Const x,y:Double; AColor:TColor);
<-------> procedure FillDemoPoints;
end;
implementation
{$R *.dfm}
{ This is the event we need to arrange Axis scale as new points are added. }
<-------> procedure TScrollForm.LineSeries1AfterAdd(S ender: TChartSeries;
ValueIndex: Longint);
begin
{ If VERTICAL SCROLL }
if CBVertical.Checked then
begin
With Sender.GetVertAxis do { <-- with the Vertical Axis... }
Begin
Automatic := False; { <-- we dont want automatic scaling }
{ In this example, we will set the Axis Minimum and Maximum values to
show One Hour of data ending at last point Time plus 5 minutes
}
Minimum := 0;
Maximum := Sender.YValues.MaxValue + DateTimeStep[dtFiveMinutes];
Minimum := Maximum - DateTimeStep[dtOneHour];
end;

ScrollBar1.Width:=Chart1.Width;
ScrollBar2.Left:=Chart1.Width- ScrollBar2.Width;
ScrollBar2.Top:=0;
ScrollBar2.Height:=Chart1.Height;
end;
end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
35
{****************************** ***************}
{ TeeChart Delphi Component Library }
{ Auto Axis scaling Demo }
{ Copyright (c) 1995-2001 by David Berneda }
{ All rights reserved }
{****************************** ***************}
unit uscroll;
interface
uses
SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
Forms, Dialogs, StdCtrls, Chart, Series, ExtCtrls, Teengine, Buttons,
TeeProcs;
type
TScrollForm = class(TForm)
Chart1: TChart;
LineSeries1: TLineSeries;
Panel1: TPanel;
Button1: TButton;
BitBtn3: TBitBtn;
CBVertical: TCheckBox;
<-------> procedure LineSeries1AfterAdd(Sender: TChartSeries;
ValueIndex: Longint);
<-------> procedure FormCreate(Sender: TObject);

```
  { fill the LineSeries with some
random data }
  LineSeries1.Clear;   { <-- this
removes all points from LineSeries1
}

  { let's add 60 minutes from 12:00 to
12:59 }
  for t:= 0 to 59 do
      AddPoint( EncodeTime( 12, t,
0,0),Random(100),clRed );

  { let's add 60 more minutes from
13:00 to 13:59 }
  for t:= 0 to 59 do
      AddPoint( EncodeTime( 13, t,
0,0),Random(100),clRed);
end;

<-------> procedure
TScrollForm.Button1Click(Sender:
TObject);
var h,m,s,msec:word;
begin
  if CBVertical.Checked then { if
VERTICAL SCROLL.... }
     DecodeTime(
LineSeries1.YValues.Last , h, m, s,
msec)
  else
     DecodeTime(
LineSeries1.XValues.Last , h, m, s,
msec);

  { add a new random point to the
Series (one more minute) }
  inc(m);
  if m=60 then
  begin
    m:=0;
    inc(h);
  end;
  AddPoint( EncodeTime( h, m, s,
msec), Random(100), clYellow );
end;

<-------> procedure
TScrollForm.CBVerticalClick(Sende
r: TObject);
```

```
  end
  else
  begin
    With Sender.GetHorizAxis do  {
<-- with the Horizontal Axis... }
    Begin
      Automatic := False;       { <-- we
dont want automatic scaling }

      { In this example, we will set the
Axis Minimum and Maximum
values to
        show One Hour of data ending
at last point Time plus 5 minutes
      }
      Minimum := 0;
      Maximum :=
Sender.XValues.MaxValue +
DateTimeStep[ dtFiveMinutes ];
      Minimum := Maximum -
DateTimeStep[ dtOneHour ];
    end;
  End;
end;

<-------> procedure
TScrollForm.AddPoint(Const
x,y:Double; AColor:TColor);
begin
  if CBVertical.Checked then { If
VERTICAL SCROLL }

LineSeries1.AddXY(y,x,'',AColor)
  else

LineSeries1.AddXY(x,y,'',AColor);
end;

<-------> procedure
TScrollForm.FormCreate(Sender:
TObject);
begin
  FillDemoPoints;
end;

<-------> procedure
TScrollForm.FillDemoPoints;
var t:Longint;
begin
```

```
ExtCtrls;

type
  TSeriesForm = class(TForm)
    GroupBox1: TGroupBox;
    BitBtn4: TBitBtn;
    BitBtn5: TBitBtn;
    BitBtn6: TBitBtn;
    BitBtn7: TBitBtn;
    GroupBox3: TGroupBox;
    BitBtn9: TBitBtn;
    BitBtn10: TBitBtn;
    BitBtn11: TBitBtn;
    BStacked: TBitBtn;
    BFastLine: TBitBtn;
    BitBtn14: TBitBtn;
    BArrowSeries: TBitBtn;
    BitBtn1: TBitBtn;
    BStackedAreas: TBitBtn;
    BitBtn21: TBitBtn;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Panel1: TPanel;
    Panel2: TPanel;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    <-------> procedure
BitBtn4Click(Sender: TObject);
    <-------> procedure
BitBtn5Click(Sender: TObject);
    <-------> procedure
BitBtn6Click(Sender: TObject);
    <-------> procedure
BitBtn7Click(Sender: TObject);
    <-------> procedure
BitBtn9Click(Sender: TObject);
    <-------> procedure
BitBtn10Click(Sender: TObject);
```

```
begin
  With LineSeries1 do
  if CBVertical.Checked then  { If
VERTICAL SCROLL }
  begin
    YValues.Order:=loAscending;
    XValues.Order:=loNone;
  end
  else
  begin
    XValues.Order:=loAscending;
    YValues.Order:=loNone;
  end;
  Chart1.LeftAxis.Automatic:=True;
{ <-- this makes axis scales
AUTOMATIC AGAIN ! }

Chart1.BottomAxis.Automatic:=Tru
e; { <-- this makes axis scales
AUTOMATIC AGAIN ! }
  FillDemoPoints;   { <-- fill sample
values again ! }
end;

end.

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
36

{******************************
***************}
{ TeeChart Delphi Component
Library          }
{ Main Form Demo
}
{ Copyright (c) 1995-2001 by David
Berneda    }
{ All rights reserved                    }
{******************************
***************}
unit udemocha;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, Buttons, StdCtrls,
```

```
   Free;
  end;
End;

<-------> procedure
TSeriesForm.BitBtn4Click(Sender:
TObject);
begin
  ShowForm(TBasicForm);
end;

<-------> procedure
TSeriesForm.BitBtn5Click(Sender:
TObject);
begin
  ShowForm(TPieForm);
end;

<-------> procedure
TSeriesForm.BitBtn6Click(Sender:
TObject);
begin
  ShowForm(TBubbleForm);
end;

<-------> procedure
TSeriesForm.BitBtn7Click(Sender:
TObject);
begin
  ShowForm(TGanttForm);
end;

<-------> procedure ShowNeedsBDE;
Begin
  ShowMessage('Needs the Borland
Database Engine.'+#13+#10+
              'Please email to get the full
TeeChart DEMO.');
end;

<-------> procedure
TSeriesForm.BitBtn9Click(Sender:
TObject);
begin
  {$IFNDEF NODB}
  ShowForm(TTablePieForm);
  {$ELSE}
  ShowNeedsBDE;
  {$ENDIF}
```

```
    <-------> procedure
BitBtn11Click(Sender: TObject);
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
BStackedClick(Sender: TObject);
    <-------> procedure
BFastLineClick(Sender: TObject);
    <-------> procedure
BitBtn14Click(Sender: TObject);
    <-------> procedure
BArrowSeriesClick(Sender:
TObject);
    <-------> procedure
BStackedAreasClick(Sender:
TObject);
    <-------> procedure
BitBtn21Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    <-------> procedure
ShowForm(AFormClass:TFormClas
s);
  end;

implementation

{$R *.dfm}
Uses
Pie,Bubble,Gantt,Basic,UStack,UFas
t,
     UArrows,StackAre,UHighLo
     {$IFNDEF NODB}

,TablePie,SQLBars,Linked,UDBHor
iz
     {$ENDIF}
     ;

<-------> procedure
TSeriesForm.ShowForm(AFormCla
ss:TFormClass);
Begin
  With AFormClass.Create(Self) do
  try
    ShowModal;
  finally
```

```
TSeriesForm.BFastLineClick(Sender: TObject);
begin
  ShowForm(TFastLineForm);
end;

<-------> procedure TSeriesForm.BitBtn14Click(Sender: TObject);
begin
  {$IFNDEF NODB}
  ShowForm(TDBHorizBarForm);
  {$ELSE}
  ShowNeedsBDE;
  {$ENDIF}
end;

<-------> procedure TSeriesForm.BArrowSeriesClick(Sender: TObject);
begin
  ShowForm(TArrowsForm);
end;

<-------> procedure TSeriesForm.BStackedAreasClick(Sender: TObject);
begin
  ShowForm(TAreasForm);
end;

<-------> procedure TSeriesForm.BitBtn21Click(Sender: TObject);
begin
  ShowForm(THighLowForm);
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
37
{*******************************************}
{    TeeChart. TChart Component  }
{ Copyright (c) 1995-2001 by David Berneda }
{    All Rights Reserved              }
{*******************************
```

```
end;

<-------> procedure TSeriesForm.BitBtn10Click(Sender: TObject);
begin
  {$IFNDEF NODB}
  ShowForm(TSQLBarsForm);
  {$ELSE}
  ShowNeedsBDE;
  {$ENDIF}
end;

<-------> procedure TSeriesForm.BitBtn11Click(Sender: TObject);
begin
  {$IFNDEF NODB}
  ShowForm(TLinkedTablesForm);
  {$ELSE}
   ShowNeedsBDE;
  {$ENDIF}
end;

<-------> procedure TSeriesForm.FormCreate(Sender: TObject);
begin
  if Screen.Width<800 then
     ShowMessage('Warning: '+#13+#10+
               'This Demo is best viewed with a Screen'+#13+#10+
               'resolution of 800x600 or greater,'+#13+#10+
               'and a Color Depth of 256 or greater.'+#13+#10+
               '16K Colors is also better than 256 Colors.');
end;

<-------> procedure TSeriesForm.BStackedClick(Sender: TObject);
begin
  ShowForm(TStackedForm);
end;

<-------> procedure
```

```
TObject);
begin
  With LineSeries1 do
  Begin
    XValues.DateTime:=False;
    Clear;
    Add( 100, 'Jan', clTeeColor );
    Add( 140, 'Feb', clTeeColor );
    Add( 120, 'Mar', clTeeColor );
    Add( 110, 'Apr', clTeeColor );
    Add(  80, 'May', clTeeColor );
    Add(  90, 'Jun', clTeeColor );
    Add(  75, 'Jul', clTeeColor );
    Add(  85, 'Aug', clTeeColor );
    Add(  95, 'Sep', clTeeColor );
    Add( 105, 'Oct', clTeeColor );
    Add( 100, 'Nov', clTeeColor );
    Add(  90, 'Dec', clTeeColor );
  end;
  With LineSeries2 do
  Begin
    XValues.DateTime:=False;
    Clear;
    Add(  60, 'Jan', clTeeColor );
    Add( 100, 'Feb', clTeeColor );
    Add(  20, 'Mar', clTeeColor );
    Add(  70, 'Apr', clTeeColor );
    AreaBrush:=bsCross;
  end;
  With ChartShape1 do
  begin
    X0:=   2;
    X1:=   8;
    Y0:= 100;
    Y1:= 130;
  end;
  With ChartShape2 do
  begin
    X0:=   5;
    X1:=   9;
    Y0:=  60;
    Y1:=  80;
  end;
  With ChartShape3 do
  begin
    X0:=   1;
    X1:=   4;
    Y0:= 135;
```

```
************}
unit ushapes;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs,TeeShape, Chart,
Series, ExtCtrls, StdCtrls, Teengine,
  Buttons, TeeProcs;

type
  TShapesForm = class(TForm)
    Chart1: TChart;
    LineSeries1: TLineSeries;
    ChartShape1: TChartShape;
    LineSeries2: TLineSeries;
    ChartShape2: TChartShape;
    Panel1: TPanel;
    Label1: TLabel;
    Label2: TLabel;
    ChartShape3: TChartShape;
    Panel2: TPanel;
    BitBtn2: TBitBtn;
    Memo1: TMemo;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
ChartShape3Click(Sender:
TChartSeries; ValueIndex: Integer;
      Button: TMouseButton; Shift:
TShiftState; X, Y: Integer);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  ShapesForm: TShapesForm;

implementation

{$R *.dfm}

<-------> procedure
TShapesForm.FormCreate(Sender:
```

```
   DataSource1: TDataSource;
   Panel1: TPanel;
   BitBtn1: TBitBtn;
   BarSeries1: TBarSeries;
   Query1: TQuery;
   Panel2: TPanel;
   Memo1: TMemo;
   BitBtn2: TBitBtn;
   ComboBox1: TComboBox;
   Label1: TLabel;
   CBRandomBar: TCheckBox;
   Panel3: TPanel;
   DBGrid1: TDBGrid;
   Label2: TLabel;
   <-------> procedure
BitBtn2Click(Sender: TObject);
   <-------> procedure
ComboBox1Change(Sender:
TObject);
   <-------> procedure
FormCreate(Sender: TObject);
   <-------> procedure
DBChart1ClickLegend(Sender:
TCustomChart;
     Button: TMouseButton; Shift:
TShiftState; X, Y: Integer);
   <-------> procedure
BarSeries1Click(Sender:
TChartSeries; ValueIndex: Integer;
     Button: TMouseButton; Shift:
TShiftState; X, Y: Integer);
   <-------> procedure
BarSeries1GetBarStyle(Sender:
TCustomBarSeries;
     ValueIndex: Longint; var
TheBarStyle: TBarStyle);
   <-------> procedure
CBRandomBarClick(Sender:
TObject);
  private
   { Private declarations }
  public
   { Public declarations }
  end;

implementation

{$R *.dfm}
Uses UDemUtil;
```

```
   Y1:= 120;
  end;
end;

<-------> procedure
TShapesForm.ChartShape3Click(Se
nder: TChartSeries;
  ValueIndex: Integer; Button:
TMouseButton; Shift: TShiftState;
X,
  Y: Integer);
begin
  ShowMessage('You clicked the
ChartShape3 (Rectangle)');
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\\\\\\\\\\

38

{*******************************
************}
{ TeeChart Delphi Component
Library       }
{ SQL Query Chart Demo
}
{ Copyright (c) 1995-2001 by David
Berneda }
{ All rights reserved                }
{*******************************
************}
unit sqlbars;

interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, StdCtrls, Buttons,
ExtCtrls, Grids, DBGrids, DB,
  DBTables, Chart, Series, DbChart,
Teengine, TeeProcs;

type
  TSQLBarsForm = class(TForm)
   DBChart1: TDBChart;
```

```
nder: TChartSeries;
  ValueIndex: Integer; Button: TMouseButton; Shift: TShiftState; X,
  Y: Integer);
begin
  With BarSeries1 do

ValueColor[ValueIndex]:=EditColor(Self,ValueColor[ValueIndex]);
end;

<-------> procedure TSQLBarsForm.BarSeries1GetBarStyle(Sender: TCustomBarSeries;
  ValueIndex: Longint; var TheBarStyle: TBarStyle);
begin
  if CBRandomBar.Checked then

TheBarStyle:=TBarStyle(Random(1+Ord(High(TBarStyle))));
end;

<-------> procedure TSQLBarsForm.CBRandomBarClick(Sender: TObject);
begin
  ComboBox1.Enabled:=not CBRandomBar.Checked;
  DBChart1.Repaint;
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
38
{**********************************************}
{ TeeChart Delphi Component Library          }
{ Stacked Bar Series Example
}
{ Copyright (c) 1995-2001 by David Berneda    }
{ All rights reserved                    }
{**********************************************}
unit ustack;

```

```

<-------> procedure TSQLBarsForm.BitBtn2Click(Sender: TObject);
begin { rerun the SQL query }
  Screen.Cursor:=crHourGlass;
  try
    Query1.Close;
    Query1.Sql:=Memo1.Lines;
    Query1.Open;
  finally
    Screen.Cursor:=crDefault;
  end;
end;

<-------> procedure TSQLBarsForm.ComboBox1Change(Sender: TObject);
begin
  if BarSeries1 is TCustomBarSeries then

BarSeries1.BarStyle:=TBarStyle(ComboBox1.ItemIndex); { <-- change bar style }
end;

<-------> procedure TSQLBarsForm.FormCreate(Sender: TObject);
begin

ComboBox1.ItemIndex:=Ord(BarSeries1.BarStyle); { <-- set combobox1 }
end;

<-------> procedure TSQLBarsForm.DBChart1ClickLegend(Sender: TCustomChart;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  With DBChart1.Legend do Color:=EditColor(Self,Color);
end;

<-------> procedure TSQLBarsForm.BarSeries1Click(Se
```

```
interface

uses
  SysUtils, WinTypes, WinProcs,
Messages, Classes, Graphics,
Controls,
  Forms, Dialogs, Chart, Series,
ExtCtrls, StdCtrls, Teengine,
ArrowCha,
  Buttons, teeprocs;

type
  TStackedForm = class(TForm)
    Chart1: TChart;
    BarSeries1: TBarSeries;
    BarSeries2: TBarSeries;
    RadioGroup1: TRadioGroup;
    Panel1: TPanel;
    Button1: TButton;
    RadioGroup2: TRadioGroup;
    ComboBox1: TComboBox;
    Label1: TLabel;
    CheckBox1: TCheckBox;
    Timer1: TTimer;
    BarSeries3: TBarSeries;
    Shape1: TShape;
    BitBtn3: TBitBtn;
    CheckBox2: TCheckBox;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
RadioGroup1Click(Sender:
TObject);
    <-------> procedure
Button1Click(Sender: TObject);
    <-------> procedure
ComboBox1Change(Sender:
TObject);
    <-------> procedure
RadioGroup2Click(Sender:
TObject);
    <-------> procedure
CheckBox1Click(Sender: TObject);
    <-------> procedure
Timer1Timer(Sender: TObject);
    <-------> procedure
Shape1MouseUp(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y:
Integer);
    <-------> procedure
CheckBox2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    <-------> procedure RefreshShape;
  end;

implementation

{$R *.dfm}
Uses UDemUtil,tecanvas;

<-------> procedure
TStackedForm.FormCreate(Sender:
TObject);
var t:Longint;
begin
  { The Chart gradient is visible only
with 16k Colors or greater video
mode }

Chart1.Gradient.Visible:=Chart1.Is
ScreenHighColor;

  { Add some random points to the
three bar Series }
  Randomize;
  with BarSeries1 do
    for t:=1 to 12 do Add(
Random(100),ShortMonthNames[t],
clTeeColor);
  with BarSeries2 do
    for t:=1 to 12 do Add(
Random(100),ShortMonthNames[t],
clTeeColor);
  with BarSeries3 do
    for t:=1 to 12 do Add(
Random(100),ShortMonthNames[t],
clTeeColor);

  { Fill a ComboBox with the three
Series titles }
  ComboBox1.Items.Clear;
  for t:=0 to Chart1.SeriesCount-1 do

ComboBox1.Items.Add(Chart1.Serie
```

```
marks }
   SeriesDown(Series[0]);
{ move series 0 to back }
   Series[SeriesCount-1].Marks.Visible:=True;   { show marks again }
  end;
  ComboBox1Change(Self);
end;

<-------> procedure TStackedForm.ComboBox1Change(Sender: TObject);
begin
  { Change the active Bar Series }
  With Chart1.Series[ComboBox1.ItemIndex] as TBarSeries do
  RadioGroup2.Itemindex:=Ord(BarStyle);
  RefreshShape;
end;

<-------> procedure TStackedForm.RadioGroup2Click(Sender: TObject);
begin
  { Change the active Bar Series Style }
  With Chart1.Series[ComboBox1.ItemIndex] as TBarSeries do
  BarStyle:=TBarStyle(RadioGroup2.Itemindex);
end;

<-------> procedure TStackedForm.CheckBox1Click(Sender: TObject);
begin
  { Start / Stop Animation }
  Timer1.Enabled:=CheckBox1.Checked;
  RadioGroup1.Enabled:=not Timer1.Enabled;
  RadioGroup2.Enabled:=not
```

```
s[t].Name);
  ComboBox1.ItemIndex:=0;

  RefreshShape;
end;

<-------> procedure TStackedForm.RefreshShape;
Begin
  { This method sets the Shape color to the active Series color in Combobox }
  With Chart1.Series[ComboBox1.ItemIndex] do
  Begin
    Shape1.Visible:=not ColorEachPoint;
    if Shape1.Visible then Shape1.Brush.Color:=SeriesColor;
  end;
end;

<-------> procedure TStackedForm.RadioGroup1Click(Sender: TObject);
begin
  { Four ways to plot bar Series: }
  Case RadioGroup1.ItemIndex of
    0: BarSeries1.MultiBar:=mbNone;
    1: BarSeries1.MultiBar:=mbSide;
    2: BarSeries1.MultiBar:=mbStacked;
    3: BarSeries1.MultiBar:=mbStacked100;
  end;
end;

<-------> procedure TStackedForm.Button1Click(Sender: TObject);
begin
  { Scroll two Bar Series }
  With Chart1 do
  Begin
    Series[SeriesCount-1].Marks.Visible:=False;  { hide
```

```
  end;

  { this function changes randomly
the Brush parameter }
  <-------> procedure
RandomBrush(ABrush:TBrush);
  Begin
    With ABrush do
    Case Random(2) of
      0: Color:=RandomColor(True);
      1: if Random(10)<5 then
Style:=TBrushStyle(Random(8));
    end;
  end;

  { this function changes randomly
the Font parameter }
  <-------> procedure
RandomFont(AFont:TFont);
  Begin
    With AFont do
    Case Random(2) of
      0: Color:=RandomColor(True);
      1: if Random(2)=0 then
            Size:=Size+1
         else
         if Size>7 then Size:=Size-1;
    end;
  End;

var tmpSeries:TBarSeries;
    t:Longint;
begin
  { This long.... routine.....
    is only a random generator for
Chart and Series properties.
  }
  Timer1.Enabled:=False; { stop
timer }

  { Set all Series.Active }
  for t:=0 to Chart1.SeriesCount-1 do
Chart1.Series[t].Active:=True;

  { Choose a Series randomly }

tmpSeries:=Chart1.Series[Random(
ComboBox1.Items.Count)] as
TBarSeries;
```

```
Timer1.Enabled;
  if not Timer1.Enabled then
ComboBox1Change(Self);
end;

<-------> procedure
TStackedForm.Timer1Timer(Sende
r: TObject);

  { this function returns a random
color from ColorPalette }
  Function
RandomColor(CheckBack:Boolean):
TColor;
  Begin
    Repeat

result:=ColorPalette[1+Random(Ma
xDefaultColors)];
    Until (not CheckBack) or
          ( (result<>Chart1.Color) and
(result<>Chart1.BackColor) );
  end;

  { this function returns a random
angle from 0, 90, 180 or 270 }
  Function RandomAngle:Integer;
  Begin
    Case Random(4) of
      0: result:=0;
      1: result:=90;
      2: result:=180;
    else
{     3: } result:=270;
    end;
  end;

  { this function changes randomly
the Pen parameter }
  <-------> procedure
RandomPen(APen:TChartPen);
  Begin
    With APen do
    if Visible then
    Case Random(3) of
      0: Color:=RandomColor(True);
      1: Style:=TPenStyle(Random(5));
      2: Width:=1+Random(3);
    end;
```

{ Then, lets change this chart a little.... }
Case Random(72) of
0: RadioGroup1.ItemIndex:=Random(RadioGroup1.Items.Count);
1,2,3,4,5: RadioGroup2.ItemIndex:=Random(RadioGroup2.Items.Count);
6: Button1Click(Self);
7: tmpSeries.SeriesColor:=RandomColor(False);
8: Chart1.Chart3dPercent:=5+Random(80);
9: Chart1.BackColor:=RandomColor(False);
10: if Chart1.Legend.Alignment=laRight then
Chart1.Legend.Alignment:=laLeft
else
Chart1.Legend.Alignment:=laRight;
11: RandomFont(Chart1.BottomAxis.LabelsFont);
12: RandomFont(Chart1.LeftAxis.LabelsFont);
13: Chart1.View3d:=not Chart1.View3d;
14: tmpSeries.BarWidthPercent:=50+Random(40);
15: RandomFont(tmpSeries.Marks.Font);
16: Chart1.BottomAxis.Grid.Visible:=not Chart1.BottomAxis.Grid.Visible;
17: Chart1.LeftAxis.Grid.Visible:=not Chart1.LeftAxis.Grid.Visible;
18:

RandomFont(Chart1.LeftAxis.Title.Font);
19: RandomFont(Chart1.BottomAxis.Title.Font);
20: tmpSeries.Marks.Style:=TSeriesMarksStyle(Random(9));
21: With Chart1 do
Begin
BevelWidth:=1+Random(10);
MarginTop :=TeeDefVerticalMargin+BevelWidth;
MarginLeft :=TeeDefHorizMargin+BevelWidth;
MarginRight :=TeeDefHorizMargin+BevelWidth;
MarginBottom:=TeeDefVerticalMargin+BevelWidth;
end;
22: Chart1.Legend.Visible:=not Chart1.Legend.Visible;
23: if Random(10)<5 then Chart1.Legend.LegendStyle:=lsSeries
else Chart1.Legend.LegendStyle:=lsValues;
24: tmpSeries.Active:=not tmpSeries.Active;
25: tmpSeries.Marks.BackColor:=RandomColor(True);
26: tmpSeries.Marks.Visible:=not tmpSeries.Marks.Visible;
27: RandomPen(Chart1.BottomAxis.Grid);
28: RandomPen(Chart1.LeftAxis.Grid);
29: Chart1.Legend.Frame.Visible:=not Chart1.Legend.Frame.Visible;
30: RandomPen(Chart1.Legend.Frame);
31:

```
RandomPen(tmpSeries.BarPen);
 32:
tmpSeries.ColorEachPoint:=not
tmpSeries.ColorEachPoint;
 33:
Chart1.Title.Alignment:=TAlignmen
t(Random(3));
 34:
RandomFont(Chart1.Title.Font);
 35:
RandomFont(Chart1.Legend.Font);
 36:
Chart1.Legend.Color:=RandomColo
r(False);
 37:
Chart1.Legend.TextStyle:=TLegend
TextStyle(Random(5));
 38:
Chart1.Legend.TopPos:=5+Random
(90);
 39:
RandomPen(Chart1.BottomAxis.Axi
s);
 40:
RandomPen(Chart1.LeftAxis.Axis);
 41:
RandomPen(Chart1.BottomAxis.Tic
ks);
 42:
RandomPen(Chart1.LeftAxis.Ticks)
;
 43:
RandomPen(Chart1.BottomAxis.Tic
ksInner);
 44:
RandomPen(Chart1.LeftAxis.TicksI
nner);
 45:
Chart1.BottomAxis.TickLength:=Ra
ndom(8);
 46:
Chart1.LeftAxis.TickLength:=Rand
om(8);
 47:
Chart1.BottomAxis.TickInnerLengt
h:=Random(8);
 48:
Chart1.LeftAxis.TickInnerLength:=
Random(8);
 49:
RandomBrush(tmpSeries.BarBrush)
;
 50:
Chart1.Title.Frame.Visible:=not
Chart1.Title.Frame.Visible;
 51:
RandomPen(Chart1.Title.Frame);
 52:
Chart1.Foot.Frame.Visible:=not
Chart1.Foot.Frame.Visible;
 53:
RandomPen(Chart1.Foot.Frame);
 54:
Chart1.Title.AdjustFrame:=not
Chart1.Title.AdjustFrame;
 55:
Chart1.Foot.AdjustFrame:=not
Chart1.Foot.AdjustFrame;
 56:
RandomBrush(Chart1.Title.Brush);
 57:
RandomBrush(Chart1.Foot.Brush);
 58:
Chart1.BottomAxis.MinorTickLengt
h:=Random(8);
 59:
Chart1.LeftAxis.MinorTickLength:
=Random(8);
 60:
RandomPen(Chart1.BottomAxis.Mi
norTicks);
 61:
RandomPen(Chart1.LeftAxis.Minor
Ticks);
 62:
RandomPen(Chart1.LeftWall.Pen);
 63:
RandomPen(Chart1.BottomWall.Pe
n);
 64:
Chart1.LeftWall.Color:=RandomCo
lor(False);
 65:
Chart1.BottomWall.Color:=Random
Color(False);
 66:
Chart1.LeftAxis.MinorTickCount:=
1+Random(6);
```

39
{**}
{ TeeChart Delphi Component Library }
{ Table Data-Aware Chart Demo }
{ Copyright (c) 1995-2001 by David Berneda }
{ All rights reserved }
{**}
unit tablepie;
interface
uses
SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
Forms, Dialogs, StdCtrls, Buttons, ExtCtrls, Grids, DBGrids, DB,
DBTables, Chart, Series, DbChart, Teengine, TeeProcs;
type
TTablePieForm = class(TForm)
DBChart1: TDBChart;
PieSeries1: TPieSeries;
Table1: TTable;
DataSource1: TDataSource;
DBGrid1: TDBGrid;
Panel1: TPanel;
Table1NAME: TStringField;
Table1SIZE: TSmallintField;
Table1WEIGHT: TSmallintField;
CheckBox1: TCheckBox;
RadioGroup1: TRadioGroup;
BitBtn1: TBitBtn;
<-------> procedure CheckBox1Click(Sender: TObject);
<-------> procedure RadioGroup1Click(Sender: TObject);
<-------> procedure PieSeries1Click(Sender: TChartSeries; ValueIndex: Integer;
Button: TMouseButton; Shift:

67: Chart1.BottomAxis.MinorTickCount:=1+Random(6);
68: Chart1.LeftAxis.Title.Angle:=RandomAngle;
69: Chart1.BottomAxis.Title.Angle:=RandomAngle;
70: Chart1.LeftAxis.LabelsAngle:=RandomAngle;
71: Chart1.BottomAxis.LabelsAngle:=RandomAngle;
end;
{ re-start timer }
Timer1.Enabled:=True;
end;
<-------> procedure TStackedForm.Shape1MouseUp(Sender: TObject; Button: TMouseButton;
Shift: TShiftState; X, Y: Integer);
begin
{ Run the Color dialog to change Series color }
With Chart1.Series[ComboBox1.ItemIndex] do
SeriesColor:=EditColor(Self,SeriesColor);
RefreshShape;
end;
<-------> procedure TStackedForm.CheckBox2Click(Sender: TObject);
begin
{ Turn on / off Chart 3D }
Chart1.View3D:=CheckBox2.Checked;
end;
end.
\\\

```
  Y: Integer);
Const CarriageReturn=#13#10;
begin
  { On Clicked Pie, show the user the pie data }
  With PieSeries1 do
  ShowMessage( XLabel[ValueIndex] + CarriageReturn+

MarkPercent(ValueIndex,True) );
end;

end.
\\\\\\\\\\\\\\\\\\\\
40
{*********************************************}
{ TeeChart Delphi Component Library 3.0       }
{ Demo                                        }
{ Copyright (c) 1995-2001 by David Berneda    }
{ All rights reserved                         }
{*********************************************}
unit teemain;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, ExtCtrls, TeEngine, Series, TeeProcs, Chart, StdCtrls,
  TeeFunci;

type
  TTeeMainForm = class(TForm)
    Panel1: TPanel;
    Button1: TButton;
    Image1: TImage;
    Timer1: TTimer;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Button6: TButton;
    Chart2: TChart;
```

```
TShiftState; X, Y: Integer);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

implementation

{$R *.dfm}

<-------> procedure TTablePieForm.CheckBox1Click(Sender: TObject);
begin

Table1.Active:=CheckBox1.Checked ;  { <-- open / close the table }
end;

<-------> procedure TTablePieForm.RadioGroup1Click(Sender: TObject);
begin
  { change the Pie value source (the table field name) }
  Case RadioGroup1.ItemIndex of
    0:
PieSeries1.PieValues.ValueSource:='Size';
    1:
PieSeries1.PieValues.ValueSource:='Weight';
  end;
  { change the Chart Foot text to display the current table Field }

DBChart1.Foot.Text[1]:='PieValueSource:=Table1'+

PieSeries1.PieValues.ValueSource;
end;

<-------> procedure TTablePieForm.PieSeries1Click(Sender: TChartSeries;
  ValueIndex: Integer; Button: TMouseButton; Shift: TShiftState; X,
```

```
Button4Click(Sender: TObject);
   <-------> procedure
Button6Click(Sender: TObject);
   <-------> procedure
Label3Click(Sender: TObject);
  private
    { Private declarations }
    DeltaZoom:Integer;
  public
    { Public declarations }
    <-------> procedure
ShowForm(FormClass:
TFormClass);
  end;

var
  TeeMainForm: TTeeMainForm;

implementation

{$R *.dfm}

Uses TeeAbout, TeeBasic, Features,
UDemoCha, Specs, ShellAPI;

<-------> procedure
TTeeMainForm.ShowForm(FormCl
ass: TFormClass);
begin
  Timer1.Enabled:=False;
  Timer2.Enabled:=False;
  With FormClass.Create(Self) do
  try
    ShowModal;
  finally
    Free;
  end;
  Timer2.Enabled:=True;
  Timer1.Enabled:=True;
end;

<-------> procedure
TTeeMainForm.FormCreate(Sender
: TObject);
begin
  TeeEraseBack:=False;
  PieSeries6.FillSampleValues(5);
  PieSeries9.FillSampleValues(5);
  PieSeries7.FillSampleValues(5);
```

```
    FastLineSeries1: TFastLineSeries;
    FastLineSeries2: TFastLineSeries;
    Timer2: TTimer;
    Label11: TLabel;
    Label12: TLabel;
    Panel2: TPanel;
    Chart1: TChart;
    Series1: TPieSeries;
    Chart7: TChart;
    PieSeries6: TPieSeries;
    Chart8: TChart;
    PieSeries7: TPieSeries;
    Chart9: TChart;
    PieSeries8: TPieSeries;
    Chart10: TChart;
    PieSeries9: TPieSeries;
    Label13: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    Label2: TLabel;
    Panel3: TPanel;
    Label1: TLabel;
    Label9: TLabel;
    Label3: TLabel;
    <-------> procedure
FormCreate(Sender: TObject);
    <-------> procedure
Timer1Timer(Sender: TObject);
    <-------> procedure
Button1Click(Sender: TObject);
    <-------> procedure
Timer2Timer(Sender: TObject);
    <-------> procedure
Button2Click(Sender: TObject);
    <-------> procedure
Chart2MouseUp(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y:
Integer);
    <-------> procedure
Chart2MouseDown(Sender:
TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y:
Integer);
    <-------> procedure
Label10Click(Sender: TObject);
    <-------> procedure
Button3Click(Sender: TObject);
    <-------> procedure
```

```
TTeeMainForm.Timer2Timer(Sender: TObject);
var tmpX:Double;
begin
  With FastLineSeries1 do
  begin
    tmpX:=XValues[1]-XValues[0];
    Delete(0);
    AddXY(XValues.Last+tmpX,Random(100),'',clTeeColor);
  end;
  With FastLineSeries2 do
  begin
    tmpX:=XValues[1]-XValues[0];
    Delete(0);
    AddXY(XValues.Last+tmpX,Random(100),'',clTeeColor);
  end;
end;

<-------> procedure TTeeMainForm.Button2Click(Sender: TObject);
begin
  ShowForm(TDemoForm);
end;

<-------> procedure TTeeMainForm.Chart2MouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  Label10Click(Self);
end;

<-------> procedure TTeeMainForm.Chart2MouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  Chart2.BevelOuter:=bvLowered;
end;

<-------> procedure TTeeMainForm.Label10Click(Sende
```

```
  Series1.FillSampleValues(5);
  PieSeries8.CheckDataSource;
  FastLineSeries1.FillSampleValues(20);
  FastLineSeries2.FillSampleValues(20);
  DeltaZoom:=5;
end;

<-------> procedure TTeeMainForm.Timer1Timer(Sender: TObject);
var tmp:Integer;
begin
  PieSeries6.Rotate(5);
  PieSeries7.Rotate(5);
  PieSeries9.Rotate(5);
  Series1.Rotate(355);
  PieSeries8.Rotate(355);
  With Chart9.View3DOptions do
  begin
    Zoom:=Zoom-DeltaZoom;
    if (Zoom<60) or (Zoom>110) then DeltaZoom:=-DeltaZoom;
  end;
  With PieSeries8 do
  begin
    tmp:=Random(Count);
    if ExplodedSlice[tmp]>20 then
      ExplodedSlice[tmp]:=ExplodedSlice[tmp]-1
    else
      ExplodedSlice[tmp]:=ExplodedSlice[tmp]+1;
  end;
end;

<-------> procedure TTeeMainForm.Button1Click(Sender: TObject);
begin
  Close;
end;

<-------> procedure
```

```
r: TObject);
begin
  ShowForm(TTeeAboutForm);
  Chart2.BevelOuter:=bvRaised;
end;

<-------> procedure TTeeMainForm.Button3Click(Sender: TObject);
begin
  ShowForm(TFeaturesForm);
end;

<-------> procedure TTeeMainForm.Button4Click(Sender: TObject);
begin
  ShowForm(TChartSpecs);
end;

<-------> procedure TTeeMainForm.Button6Click(Sender: TObject);
begin
  ShowForm(TSeriesForm);
end;

<-------> procedure TTeeMainForm.Label3Click(Sender: TObject);
Var St:Array[0..255] of char;
begin
  Timer1.Enabled:=False;
  Timer2.Enabled:=False;

ShellExecute(0,'open',StrPCopy(St,'http://'+Label3.Caption+

'/products/teechart/delphi6'),nil,nil,SW_SHOW);
end;

end.
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\\\\\\\\\\\
The end
 ALAA EDDIN  LUBBAD
00963944575371
```